YOUMU BOOKSTORE

PART.3 DESIGN MODEL

2019.6.7 | by: Group 3

Table of Contents

1. Overview

1.1. Background

As the booming of network in recent years, e-commerce has been widely accepted by a variety of the Internet users. Being a part of which, online book store system is developing rapidly as well.

In a form of website or application, online book store can be taken advantage to sell books. Online book store opens one after another with the broad adaptation of e-commerce in the aspect of book selling. For one thing, a majority of people have formed the habit that shop online. The number of which has been increasing continuously. For another thing, from the aspect of shopping items, books are relatively cheap and standard, so it is of few risks to buy a book and online purchasing can be accepted easily. Comparing to the traditional book store, the online book store have a lot of advantages. As for sellers, they can not only avoid the limitation and blindness while ordering books, but also surmount the obstacles of high expanses, tough management and costly ordering. As for customers, they are capable of learning details, looking through samples and ordering books swiftly. It is difficult for offline transaction to have the above advantages.

All in all, developing an online book store system is available as well as fashionable.

1.2. Purpose & Definition

Based on the fundamental functions of online book store and our brainstorming, the participants of the online book store system involve users (visitors, registered user, senior user), administrators, publishers and third-party payment enterprises.

According to the advanced design, the online book store system can maintain the following functions:

1)As for users: have access to check and alter part of the personal information, be able to search, add to shopping cart, add to order form and ask for after-sale service, can share own reading schedule and comments.

2)As for administrators: update the details of each book, control the basic settings of the system, send messages to users in time and get touch with publishers in order to inform them to supply the stock.

3)As for publishers: receive the timely messages which come from administrators and feedback the stock information.

4)As for third-party payment enterprises: give authority to the payments in the order forms.

5)Keep limits of authority distinct so that it is conducive to the protection of data and intimacy securities.

6)Find latent users in order to expand our customers.

To coordinate with the online book store's style, we are expecting to provide a cozy environment for users to stroll in the book store at the same time of feeling the pleasure of reading besides make it possible that users are able to easily buy a book online. By using our online book store system, users can attain a positive shopping and reading experience.

Moreover, as an online book store, we are determined to endow a sense of belonging to every user by forming a unique culture. So we have designed an UI of chinoiserie and have entitled our store YOUMU Book Store.

1.3. System Overview

1.3.1. System Structure

The high level architecture is divided as four different layers which have their own certain functions, named as Presentation Layer, Services Layer, Controller Layer and Data Layer.

1.3.2. Design

The customers will send request from the browser on his/her device, then the browser accesses the targeted web pages through Uniform Resource Locator (URL), which colloquially termed a web address. After retrieving the request, the View invokes the model to retrieve information from the backend database. The tuples in the database will be returned as an object, then the view will process them, select the appropriate templates, padding with the appropriate parameters, and finally send the HTML web page to the browser. In fact, our architecture is framework based, Views and models are implemented at different levels.

1.3.3. Description of Implementations

We choose to use the popular CSS and JavaScript frameworks, such as Vue2.0 and jQuery to design our HTML web pages. jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. It is a free, open-source software using the permissive MIT license. Web analysis indicates that it is the most widely deployed JavaScript library by a large margin. Vue2.0 is a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. Based on Google's V8 engine, Node.js is an event-driven I/O server-side JavaScript environment. Simply, Node.js runs in the backend written by JavaScript. We take advantage of Express as framework to develop server and what's more, to apply to the Web application, attaching great functions to our online book store system. As for the Persistence, we use the MySQL as DBMS to maintain the data.

1.4. Progress

Since the last project, a great progress has been made. The English version is totally developed, which means English is adapted to each of our diagrams as well as word descriptions.

As for the architecture, according to the result of our brainstorming, we have refined the architecture with considering the ways of implementation based on the original 4 layers. Each layer has specific dependent platform. What's more, a label of subsystems and interfaces is listed to demonstrate the operations and interfaces clearly. And several samples are presented to show the details of our process.

As for the design mechanisms, we have searched a lot of relative reports to learn about it and make use of some of them to our online book store system. Persistence mechanism and Information Exchange mechanism are exemplified specifically with their class diagram, sequence diagram and description, including the suitable implementation platform.

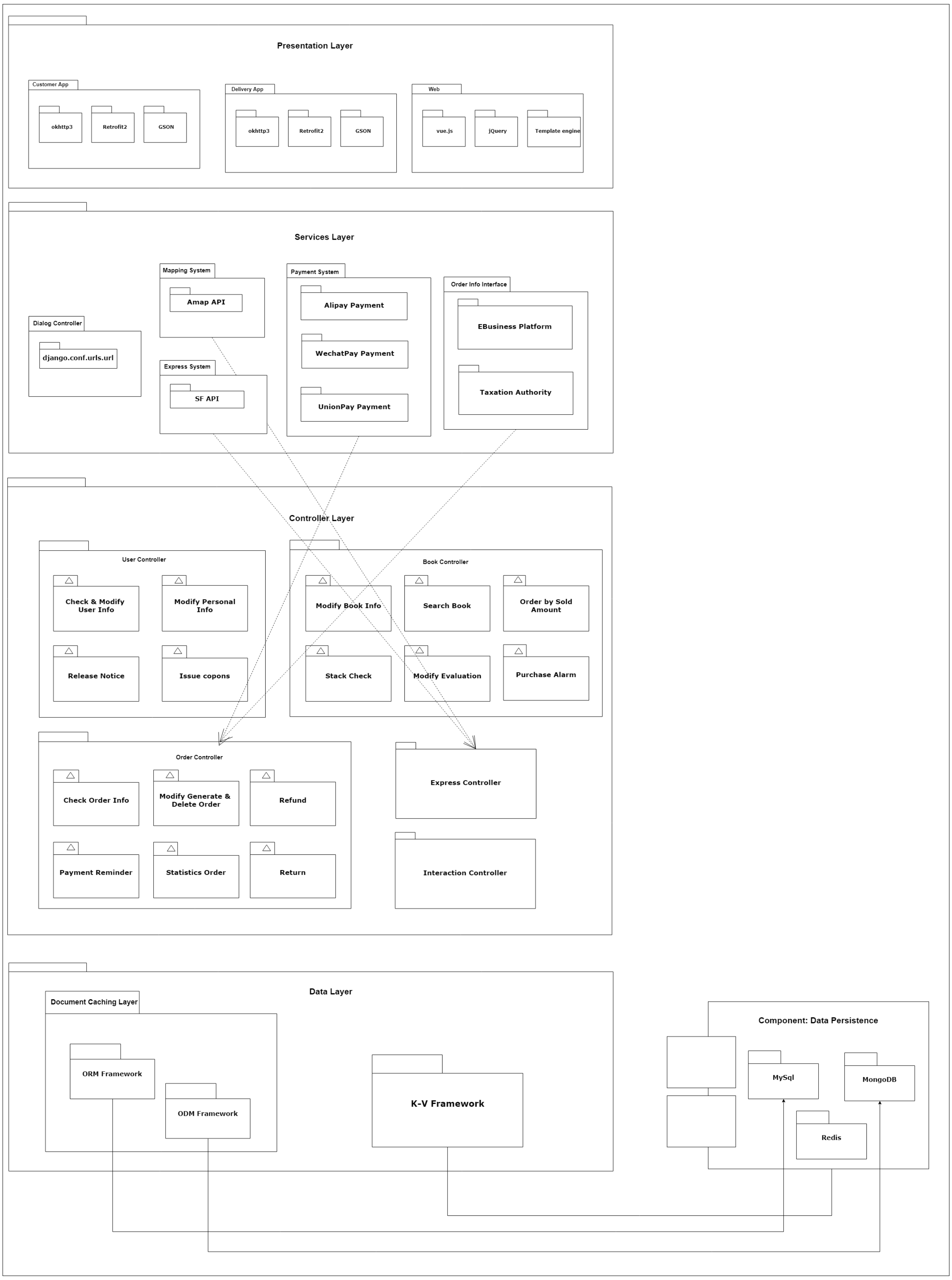
As for the use case realization, the above design mechanism is combined with the architecture, presented through the use case.

As for the prototyping, we have already created a demo connecting the database and our login subsystem interface. Users are able to register and log in though the browse. The coordinate data of him or her will be stored perpetually.

Apart from the use case model we made in advance, we analyzed the whole system and separated it to 9 subsystems, suppling the class diagram, the sequence diagram, etc. The class diagrams are updated from the previous analysis class diagrams, being suppled with functions and boundary or control classes.

As for the design mechanism, the architecture and interfaces are presented, especially the detailed interfaces showing the major pages of our online book store system. The interpretation is shown in these pages.

2. Architecture Refinement



2.1. Global Architecture

The high level architecture is divided as four different layers which have their own certain function , and also the majority of them correlative with others, they named as **Presentation Layer**, **Services Layer**, **Controller Layer** and **Data Layer**.

2.1.1. Presentation Layer

The Presentation Layer is in charge of providing user interface and handle user interactions. We renew it from the last assignment which we called it as UI Layer and the packages were also renew from the Mobile Terminal UI and Web Terminal UI.

On the web frontend, we choose the popular CSS and JavaScript frameworks, such as **vue.js** and **jQuery** to design our HTML web pages.

> Why do we choose vue.js?
>
> Vue is a progressive framework for building user interfaces. Unlike other large frames, Vue is designed to be applied from the bottom up. Vue's core library focuses only on the view layer, making it easy to get started and easy to integrate with third-party libraries or existing projects. On the other hand, Vue is also fully capable of driving complex single-page applications when used in conjunction with modern toolchains and various support libraries.
>
> Why do we choose jQuery?
>
> jQuery is a JavaScript tool library (class library) that gets a whole set of defined methods by encapsulating native JavaScript functions. Integrates the power of JavaScript, CSS, DOM and Ajax.
>
> We can write only a little code to gain a wonderful effect.

The **Template Engine** is a tool for parsing corresponding type template files and then dynamically generating view files consisting of data and static pages. It responds to various parsing actions through tags, and dynamically displays the corresponding data to a specified location by means of variable occupancy.

We have also imported the **template engine of Django** in the realization of web frontend, which serves as the templates of MTV architecture pattern.

Since our Customer APP and Delivery APP are based on Android, frameworks like okhttp3, Retrofit2, GSON will be used in those modules.

> Okhttp
>
> Mainstream web request framework.
>
> The whole process is: Convert the constructed Request to Call through OkHttpClient, then perform asynchronous or synchronous tasks in RealCall, and finally send out the network request and get the returned response through some interceptor interceptor.
>
> Why do we choose OkHttp?

1)Support http2, share all requests for one machine to share the same socket

2)Built-in connection pool, support connection multiplexing, reduce latency

3)Support transparent gzip compression response body

4)Avoid duplicate requests through caching

5)Automatically retry the host's other ip when the request fails, automatically redirect

6)Easy to use API

Retrofit2

Retrofit2 is simply a network request adapter that translates a basic Java interface into an HTTP request via a dynamic proxy and sends the request through OkHttp. It also has great scalability, support for various format conversions and RxJava.

GSON

GSON is a Java class library provided by Google to map between Java objects and JSON data. You can convert a Json character into a Java object or convert a Java to a Json string.

Why do we choose GSON?

1) fast and efficient

2) the amount of code is small, simple

3) object-oriented

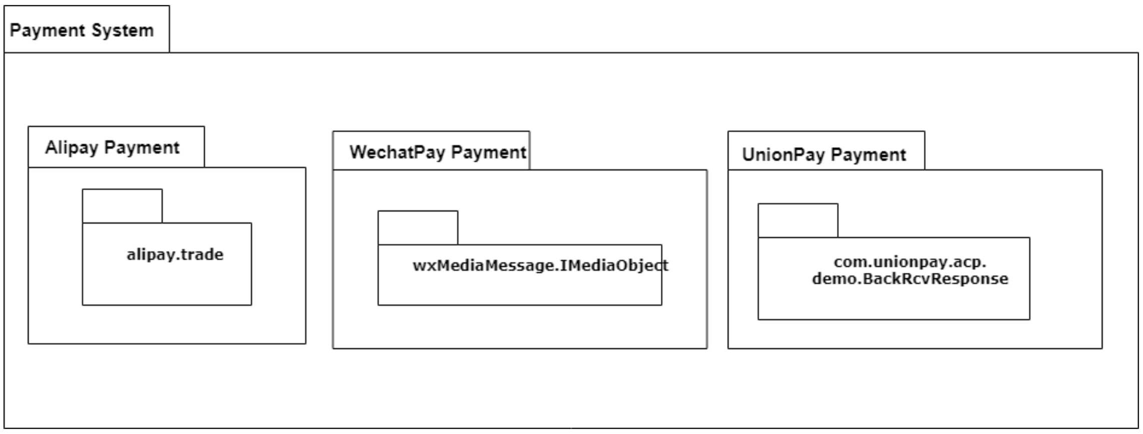
4) data transfer and analysis is convenient

2.1.2. Services Layer

It provides unified API interfaces of all functionalities for both web clients and mobile clients. The web APIs are provided in the form of JSON services based on HTTP/HTTPs.

In the Dialogue Controller subsystem, we import Django's url library **"Django.conf.urls.url"**. The **Amap API** is called in the mapping system, and the **SF API** is called in the Express System which is used for checking logistics information such as: Where the books now I bought? How far is it from me? etc.

The Payment System supports three kinds of payment modes, **Alipay**, **WechatPay** and **UnionPay**. We read the official document of the API and designed adapters for those to help us to use it conveniently.



2.1.3. Controller Layer

In this layer, we build three main controllers: User Controller, Book Controller, Order Controller. And we also hold a Delivery Schedule and Django_view.

In User Controller, it mainly contain the behavior that the users interact with the system. The user can check and modify his or her information, modify personal information, releases notices(this is only for administrator) and get issue coupons.

In Book Controller, it contain modify the book information, search book, order by sold amount, stack check(if close to the limit amount, we should send a message to the publisher), modify the evaluation and purchase alarm.

In order Controller, it contain check order information , modify generate or delete order, refund, payment reminder, statistics order and return.

We also create an Express Controller, which is powered by Amap API and SF API. It is used for customer to check his or her express information and can get the location of the books he or she brought at the first time.

2.1.4. Data Layer

The Data Access Layer is in charge of all data accessing operations. It provides unified data accessing interfaces for different data models. The system introduces three different databases for storing data:

MySQL for storing most part of the data, which is highly relational and has relatively high demand of ACID;

MongoDB for storing mass data that is less relational and has less demand of ACID, such as GPS location data from all GPS sensors in each collecting period (30 seconds). MongoDB enables those kinds of data to be stored and accessed more efficiently.

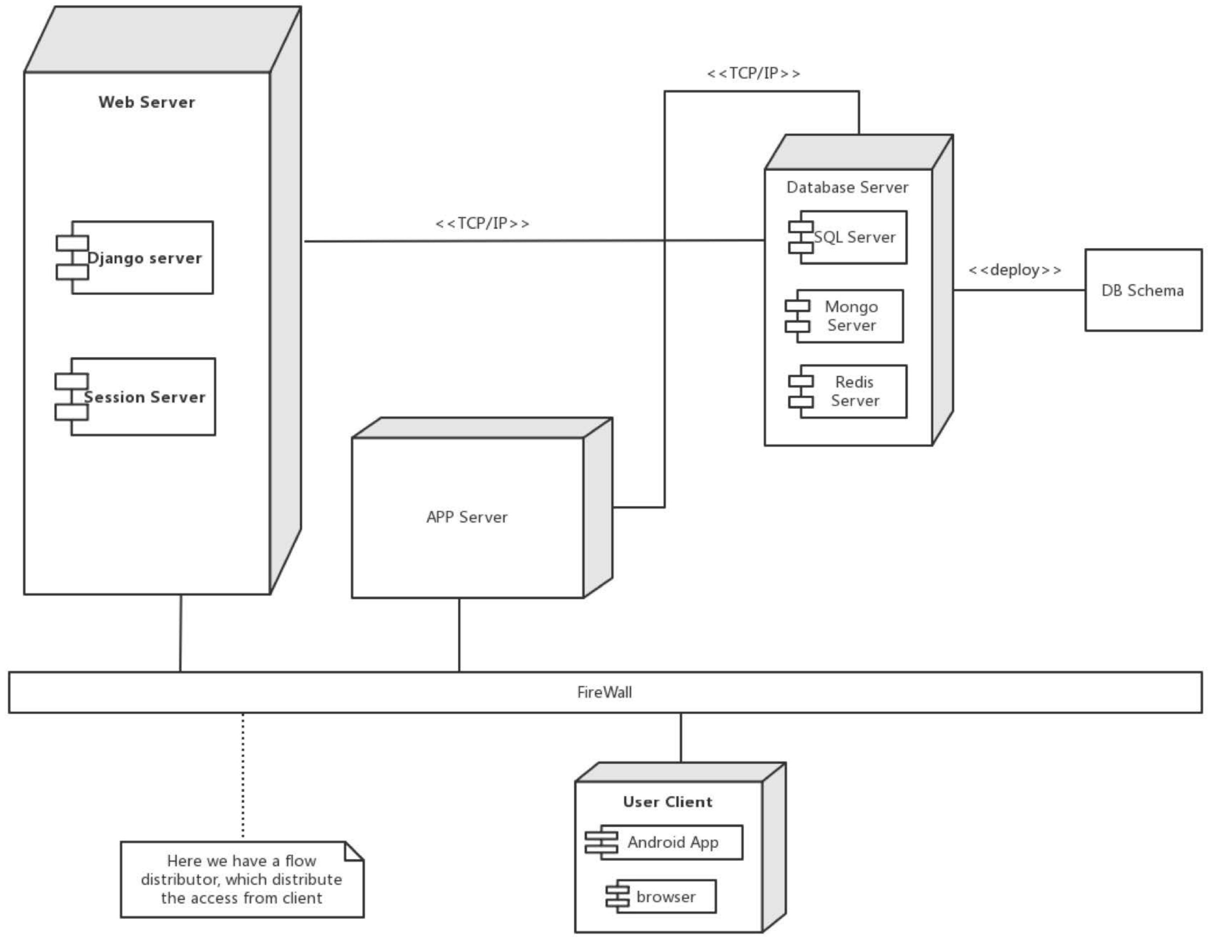
Finally, the system requires **Redis**, which is an in-memory (however data can be persistent) high performance Key-Value database, which is used to store cache and sessions that are highly performance sensitive. Notice that Redis also supports a rich kind of data structures, which can significantly boost the calculation of some kind of ranking related tasks.

For MongoDB, the system uses ODM (Object Document Mapping) frameworks to map objects to MongoDB queries.

For MySQL, the system uses ORM (Object Relational Mapping) frameworks to map objects to SQL queries.

ORM and ODM frameworks provides better readability while fundamentally prevents SQL injections. Both ODM and ORM frameworks exposes interface under the caching layer. In this kind of architecture, the caching layer is transparent to the developers. Developers need not to care about whether to access the cache or database. This can simplify the development process while keeping high performance. The cache layer is dependent on Redis (and its drivers) of course.

2.1.5. Deployment diagram



2.1.5.1. User Client

We hold two kinds of service mode for User Client: Android app and browser.They are provided services by different servers.

2.1.5.2. Firewall

There is a flow distributor on the firewall which distribute the assess from the client. All operations on the server are filtered through the firewall.

2.1.5.3. Web Server

The session server is designed for provide authentication services. Page routing and rendering are finished by Django server.

2.1.5.4. Data Server

The web server and app server share one database server, where Redis provides key-value cache and improves authentication speed. Besides, SQL server and Mongo server will assume relational and non-relational data storage services respectively.

> Why do we choose Mongo Server?
>
> MongoDB Atlas delivers the world's leading database for modern applications as a fully automated cloud service engineered and run by the same team that builds the database. Proven operational and security practices are built in, automating time-consuming administration tasks such as infrastructure provisioning, database setup, ensuring availability, global distribution, backups, and more. The easy-to-use UI and API let you spend more time building your applications and less time managing your database.
>
> Why do we choose Redis Server?
>
> Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes with radius queries and streams. Redis has built-in replication, Lua scripting, LRU eviction, transactions and different levels of on-disk persistence, and provides high availability via Redis Sentinel and automatic partitioning with Redis Cluster.

2.2. Subsystems and Interfaces

SubSystem	Provided Interfaces	Operations	Required Interfaces	Operations

APP	Customer Access	+getExpress()
		+getPosition()
		+HistorySearch(Tracking_number)
			OKHttp	+OkHttpClient()
				+client.newCall(request).execute()
				+response.isSuccessful()
				+response.body()
				+onFailure()
				+onResponse()

	Administrator Access			
Web	WEBFronted	+POST(args)		

Map	Map	+getMap(location)
		+getAddress()
		+getLocation(address)
		+getEstimateTime(locatioin,address)
			Amap API	+Map(id,option)
				+getCenter()
				+setView(center,zoom)
				+routePath(startOption,stopOption,sucess back,errback,option)

Express	Express	+getExpress()
		+getPosition()
		+HistorySearch(Tracking_number)
			SF API	+getOrderServiceRequestXml(Expres sOrder expressOrder)
				+CallExpressServiceTools.callSfExpress ServiceByCSIM(requestXml)
				+shunFengCallback(String content)

Payment	Payment	+payOnline(orderID,paymentWay, receiptType,coupon)
		+payOffline(orderID,userID,paym entWay,receiptType ,coupon)
		+getPaymentReceipt(orderID)
		+getPaymentInformation(orderID)
		+refund(orderID)
		+choosePayMethod()
		enterPassWord()
		+showPayment()
			AlipayPayment	+DefaultAlipayClient(args)
				+AlipayTradePrecreateRequest(args)
				+AlipayTradePayRequest(args)
				+AlipayTradeRefundRequest(args)

			WechatPayment	+POST(args)
				+IMediaObject(args)
				+wxpay.unifiedOrder()
				+wxpay.orderQuery()
				+wxpay.refundQuery()
				+wxpay.downloadBill()
				+WXPayUtil.mapToXml()
				+wxpay.isPayResultNotifySignatureValid()
				+WXPayUtil.xmlToMap()

			UnionPayment	+AcpService.sign()
				+SDKConfig.getConfig().getBackRequestU rl()
				+AcpService.validate()
				+rspData.get()

Order	OrderInformation	+checkPayment()
		+confirmPayment()
		+getOrderLocation()
		+createInvoice(title, items, type)
		+check(userID, signature)
		+ check(userID, fingerprint)
			EBusinessPlatform	+getNewOrder():Order

			TaxationAuthority	+createNewInvoice(Order,title ,timestamp):invoiceID +getInvoice(invoiceID):Invoice

	OrderManagement	+getOrder(orderID)
		+getOrderList(orderIDList)
		+updateOrder(orderID,commandT ype,args)
		-setWithdraw(orderID)
		-judgeReplace(order)		

User Management	Customer_Operation	+register(name,password,ID)
		+login(ID,password)
		+setInfo()
		+favourite(Book:Book)
		+selectBook(info:string)
		+immediateBuy(Book:Book)
		+setContent(content:string)
		+sendMessage(Message)		

	Administrator_Operation	+login(ID,password)
		+setInfo()
		+setContent(content:string)
		+sendMessage(Message)
		+modifyUserData()
		+modifyBookData()		

	Publisher_Operation	+register(name,password,ID)
		+login(ID,password)+setInfo()
		+setContent(content:string)
		+sendMessage(Message)		

Model	IOrderModel	+create(dict)
		+save()
		+update(dict)
		+delete(dict)
		+get(dict)
		+filter(dict)
		+sort(dict)
		+check(dict)
			ODM Framework
			ORM Framework
			K-V Framework	

	IScheduleModel			
	IAddressModel			
	IPaymentModel			
	IInvoiceModel			
	IUserModel			
	IDiscountModel			
	ITaxModel			
	IInventoryModel			

DataAccess	MySQL	+SELECT(args)
		+INSERT(args)
		+UPDATE(args)
		+DELETE(args)		

	MongoDB			
	Redis			

3. Interfaces Between Our System and External Systems

Specify the 3rd party API as well as the map services in detail.

3.1. Express Check Subsystem

If the customer want to know: Whether the books are shipped? Where the books I brought yesterday? How far is it from me? and so on. He or she can use this subsystem to know these information.

After order query, the query page will save the order's receiver's tracking number and adress in the cookie, so that the express check page can get it by reading the cookie and the query will get the express information by calling the SF API , other information for address especially the estimate time by calling the AMap API and interfaces in sequence:

3.1.1. Get customer code and check code

Apply for the api interface: https://qiao.sf-express.com/index.html

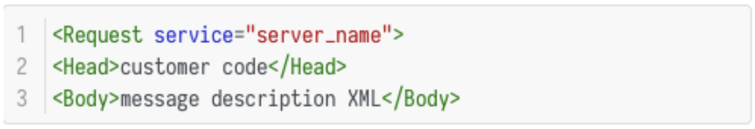
We can get the *url*, *clientCode*, *checkword*

url: http://bsp-oisp.sf-express.com/bsp-oisp/sfexpressService

3.1.2. XML message description

1) Request a XML message

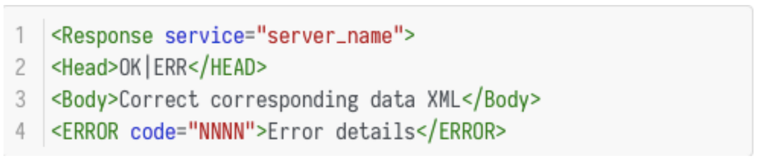
The service attribute defines the "service name"; the Head element defines the "customer code".



```
<Request service="server_name">
<Head>customer code</Head>
<Body>message description XML</Body>
```

2) Response XML message

The value of the Head element is "OK" or "ERR"; OK means the transaction is successful, ERR means that the system or business is abnormal, the transaction fails; for the batch trading scenario, it can only be success/failure, no partial success/partial failure.



```
<Response service="server_name">
<Head>OK|ERR</HEAD>
<Body>Correct corresponding data XML</Body>
<ERROR code="NNNN">Error details</ERROR>
```

3.2.

T o coordinate with the online book store’s style, we are expecting to provide a cozy environment for users to stroll in the book store at the same time of feeling the pleasure of reading besides make it possible that users are able to easily buy a book online. By using our online book store system, users can attain a positive shopping and reading experience.

If the customer want to know: Whether the books are shipped? Where the books I brought yesterday? How far is it from me? and so on. He or she can use this subsystem to know these information.

Moreover, as an online book store, we are determined to endow a sense of belonging to every user by forming a unique culture. So we have designed an UI of chinoiserie and have entitled our store YOUMU Book Store.

4.

4.1. System Structure

The high level architecture is divided as four different layers which have their own certain functions, named as Presentation Layer, Services Layer, Controller Layer and Data Layer.

4.2. Design

The customers will send request from the browser on his/her device, then the browser accesses the targeted web pages through Uniform Resource Locator (URL), which colloquially termed a web address. After retrieving the request, the View invokes the model to retrieve information from the backend database. The tuples in the database will be returned as an object, then the view will process them, select the appropriate templates, padding with the appropriate parameters, and finally send the HTML web page to the browser. In fact, our architecture is framework based, Views and models are implemented at different levels.

4.3. Description of Implementations

We choose to use the popular CSS and JavaScript frameworks, such as Vue2.0 and jQuery to design our HTML web pages. jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. It is a free, open-source software using the permissive MIT license. Web analysis indicates that it is the most widely deployed JavaScript library by a large margin. Vue2.0 is a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. Based on Google's V8 engine, Node.js is an event-driven I/O server-side JavaScript environment. Simply, Node.js runs in the backend written by JavaScript. We take advantage of Express as framework to develop server and what's more, to apply to the Web application, attaching great functions to our online book store system. As for the Persistence, we use the MySQL as DBMS to maintain the data.

5. Progress

Since the last project, a great progress has been made. The English version is totally developed, which means English is adapted to each of our diagrams as well as word descriptions.

As for the architecture, according to the result of our brainstorming, we have refined the architecture with considering the ways of implementation based on the original 4 layers. Each layer has specific dependent platform. What's more, a label of subsystems and interfaces is listed to demonstrate the operations and interfaces clearly. And several samples are presented to show the details of our process.

As for the design mechanisms, we have searched a lot of relative reports to learn about it and make use of some of them to our online book store system. Persistence mechanism and Information

Exchange mechanism are exemplified specifically with their class diagram, sequence diagram and description, including the suitable implementation platform.

As for the use case realization, the above design mechanism is combined with the architecture, presented through the use case.

As for the prototyping, we have already created a demo connecting the database and our login subsystem interface. Users are able to register and log in though the browse. The coordinate data of him or her will be stored perpetually.

Apart from the use case model we made in advance, we analyzed the whole system and separated it to 9 subsystems, suppling the class diagram, the sequence diagram, etc. The class diagrams are updated from the previous analysis class diagrams, being suppled with functions and boundary or control classes.

As for the design mechanism, the architecture and interfaces are presented, especially the detailed interfaces showing the major pages of our online book store system. The interpretation is shown in these pages.

Your Plan Title Here

1. Executive Summary

This is a summary of the key points of your business plan, as outlined in the following chapters.

2. Company Summary

Describe your company, who you are, where you operate.

2.1. Start-up Summary

Summarizes your Start-up table numbers, both expenses and assets. These occur before you are open for business.

Table: Start-up

Start-up Requirements	$0
Start-up Expenses	$0
Expense 1	$0
Expense 2	$0
Expense 3	$0
Expense 4	$0
Expense 5	$0
Expense 6	$0
Total Start-up Expenses	$0
Start-up Assets	$0
Cash Required	$0
Start-up Inventory	$0
Other Current Assets	$0
Long-Term Assets	$0
Total Assets	$0
Total Requirements	$0

3. Services

Describes the products and/or services you offer, how they are provided and by whom, and plans for future service offerings.

4. Market Analysis Summary

4.1. Describes the different groups of target customers included in your market analysis and explains why you are selecting these as targets.

Table: Market Analysis

Potential Customers	Growth	2020	2021	2022	2023	2024	CAGR
Segment Name	$0	$0	$0	$0	$0	$0	$0
Segment Name	$0	$0	$0	$0	$0	$0	$0
Segment Name	$0	$0	$0	$0	$0	$0	$0
Total	$0	$0	$0	$0	$0	$0	$0

5. Strategy and Implementation Summary

Summarizes the organizational strategy for target marketing, sales and marketing activities, and product/service development.

5.1. Sales Forecast

This topic explains the Sales Forecast table.

Table: Sales Forecast

Sales	FY 2020	FY 2021	FY 2022
Row 1	$0	$0	$0
Row 3	$0	$0	$0
Total Sales	$0	$0	$0

Direct Cost of Sales	FY 2020	FY 2021	FY 2022
Row 1	$0	$0	$0
Row 2	$0	$0	$0
Row 3	$0	$0	$0
Subtotal Direct Cost of Sales	$0	$0	$0

5.2. Milestones

Describes the milestones (measurable activities) laid out in the Milestones table.

Table: Milestones

Milestones	Start Date	End Date	Budget	Manager	Department
Name me	11/30/20	11/30/20	$0	ABC	Department
Name me	11/30/20	11/30/20	$0	ABC	Department
Name me	11/30/20	11/30/20	$0	ABC	Department
Name me	11/30/20	11/30/20	$0	ABC	Department
Name me	11/30/20	11/30/20	$0	ABC	Department
Name me	11/30/20	11/30/20	$0	ABC	Department
Name me	11/30/20	11/30/20	$0	ABC	Department
Total					$0

6. Management Summary

Describes the management and personnel structure of the company, including any gaps that need to be filled.

7. Financial Plan

Summarizes the financial aspects of your business plan.

7.1. Start-up Funding

Explains where your funding will come from, in what form (as investments and/or loans), and how this funding will cover the start-up requirements outlined in the Start-up table.

Table: Start-up Funding

Start-up Funding	
Start-up Expenses to Fund	$0
Start-up Assets to Fund	$0
Total Funding Required	$0
Assets	
Non-Cash Assets from Start-up	$0
Cash Requirements from Start-up	$0
Additional Cash Raised	$0
Cash Balance on Starting Date	$0
Total Assets	$0
Liabilities	
Current Borrowing	$0
Long-Term Liabilities	$0
Accounts Payable (Outstanding Bills)	$0
Other Current Liabilities (Interest-Free)	$0
Total Liabilities	$0
Capital	
Planned Investment	$0
Owner	$0
Investor	$0
Additional Investment Requirement	$0
Total Planned Investment	$0
Loss at Start-up (Start-up Expenses)	$0
Total Capital	$0

Total Capital and Liabilities	$0
Total Funding	$0

7.2. Projected Profit and Loss

Explains the important points of your Profit and Loss projections, such as percentage increase in sales and profits, your gross margins, and key budget items.

Table: Start-up Funding

Pro Forma Profit and Loss Sales	FY 2020	FY 2021	FY 2022
Direct Cost of Sales	$0	$0	$0
Other Cost of Sales	$0	$0	$0
Total Cost of Sales	$0	$0	$0

Gross Margin	$0	$0	$0
Gross Margin %	0%	0%	0%

Expenses			
Payroll	$0	$0	$0
Expense 2	$0	$0	$0
Depreciation	$0	$0	$0
Rent	$0	$0	$0
Utilities	$0	$0	$0
Insurance	$0	$0	$0
Payroll Taxes	$0	$0	$0
Other	$0	$0	$0

Total Operating Expenses	$0	$0	$0

Profit Before Interest and Taxes	$0	$0	$0
EBITDA	$0	$0	$0
Interest Expense	$0	$0	$0
Taxes Incurred	$0	$0	$0

Net Profit	$0	$0	$0
Net Profit/Sales	$0	$0	$0