上海海事大学

软件项目管理课程报告

课程题目： iplan 日程表设计与实现

姓　　名： 黄德玉　　学号： 202230310232

姓　　名： 谢广伟　　学号： 202230310305

姓　　名： 刘智清　　学号： 202230310242

学　　院： 信息工程学院

指导教师： 刘晋

完成时间： 2023 年 5 月

iPlan requirement specifications

Contents

1 requirements analyst

1.1 Overview of background and requirements

In daily work and life, a large part of users often use Outlook mailbox, Outlook is Microsoft's main mail transmission and collaboration client products. It is a standalone application integrated into Microsoft Office and Exchange Server. Full integration of features such as email, calendar, and contact management makes Outlook the perfect client in the eyes of many business users. However, with the increase of email and calendar, users need to open outlook mailbox frequently to check email and calendar information, and there are many information that customers do not need to deal with, including advertising information, which causes a waste of time to some extent.

iPlan is designed to solve this problem. Users can configure personal Outlook mailbox account in iPlan, regularly or manually pull Outlook mailbox mail and calendar information, and generate personal calendar after filtering and filtering. In iPlan, users can view the schedule for the day, the week or the month, and can also query the corresponding schedule by using the email address of the sender. When synchronizing the email data, the popular words are recorded according to the title of the email (the title is segmented according to the part of speech), and the top ten popular words are displayed in the calendar. Users can query the relevant email content according to the keywords. In addition to syncing data, users can also add and delete their own schedule. Schedule information can be exported to an Excel sheet for offline processing.

1.2 Flow Chart

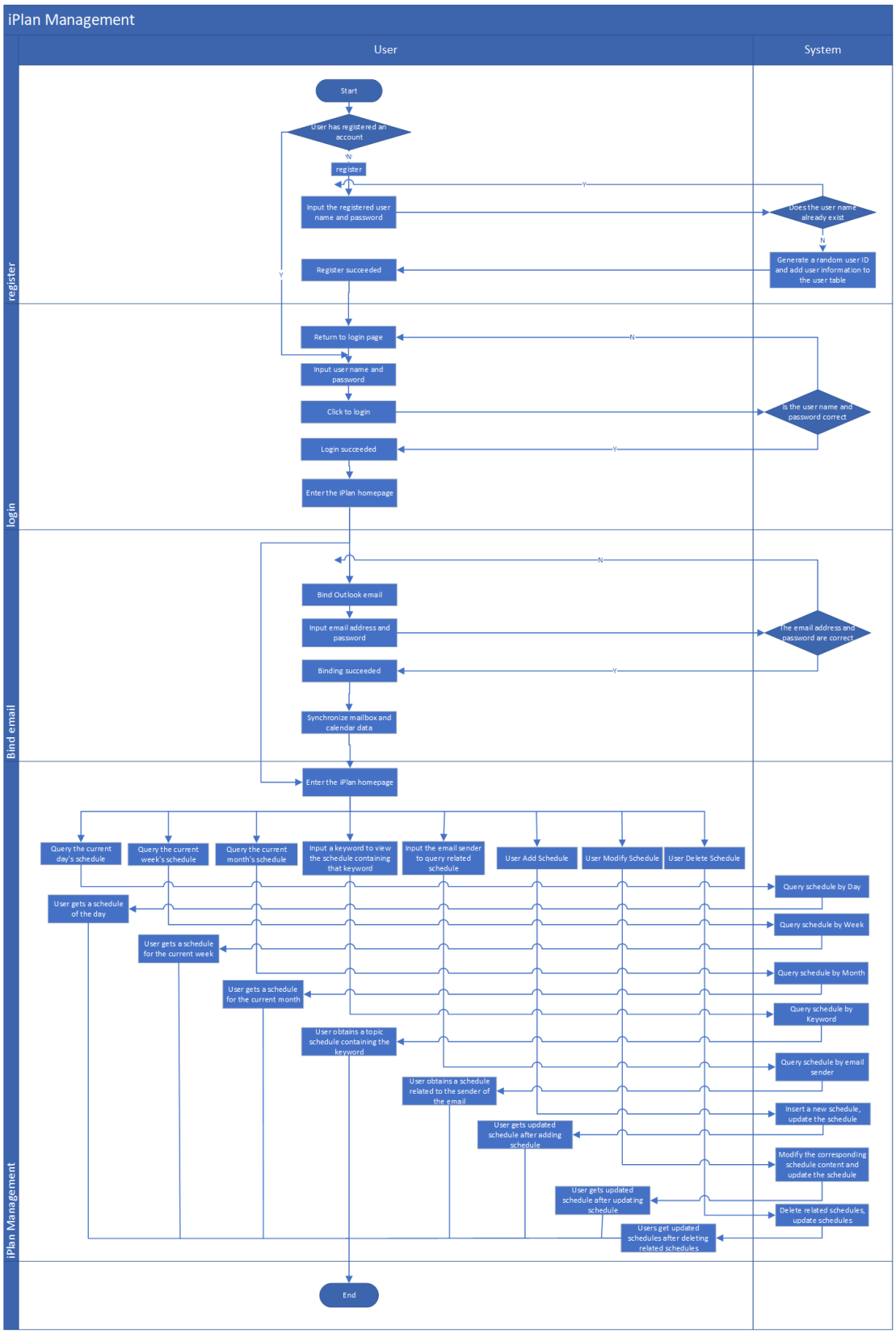
The user clicks the registration button to enter the registration page;Enter a valid user name and password to successfully register the account;

The user returns to the login interface and enters the correct user name and

password to successfully log in;Users can choose to perform schedule management or bind Outlook email;

When selecting to bind Outlook email, enter the correct email address and password to successfully bind the mailboxes, and synchronize email data and calendar meeting data; If Outlook email binding is not performed, you can directly enter the schedule management homepage;

Enter the iPlan homepage, and users can view the schedule by day, week, and month; You can also query the schedule based on keywords or the email sender; The operation of adding a schedule can be realized; You can modify and update the existing schedule content; You can delete useless schedules.

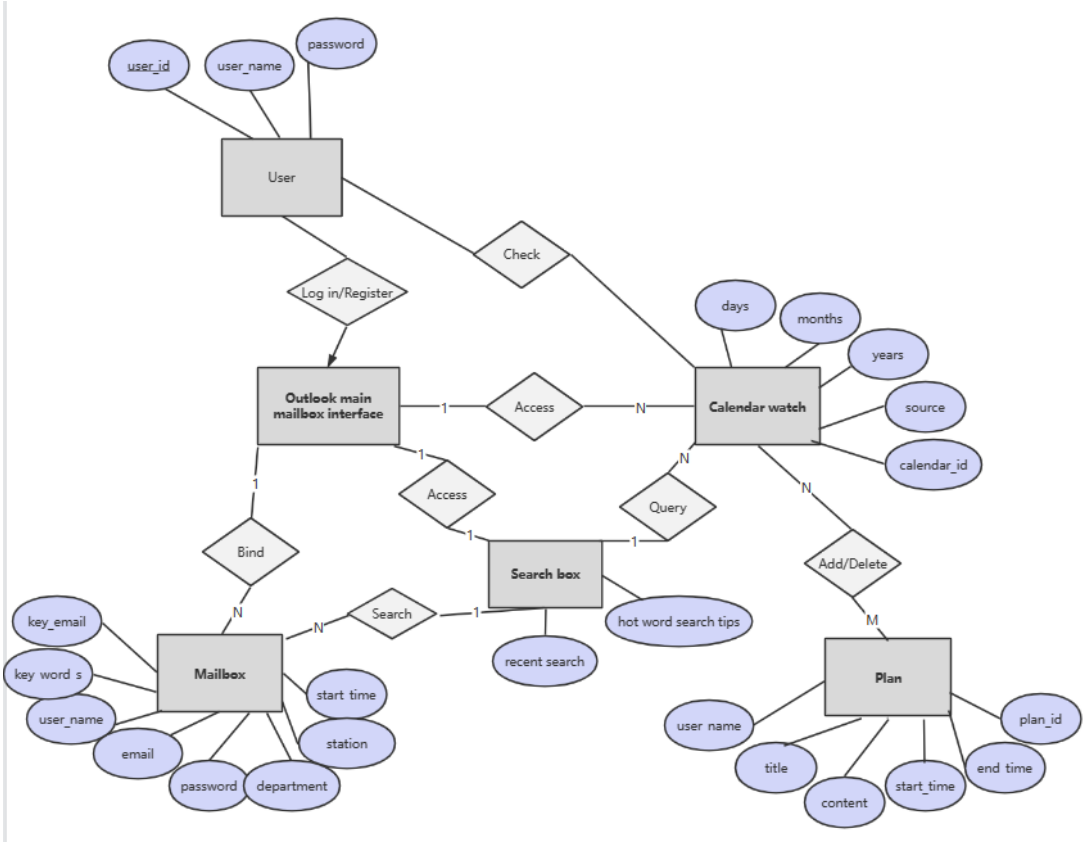

iPlan Management
User
System
register
login
Bind email
iPlan Management
Start
User has registered an account
N
register
Y
Input the registered user name and password
Does the user name already exist
N
Generate a random user ID and add user information to the user table
Register succeeded
Y
Return to login page
N
Input user name and password
Click to login
Is the user name and password correct
Login succeeded
Y
Enter the iPlan homepage
N
Bind Outlook email
Input email address and password
The email address and password are correct
Binding succeeded
Y
Synchronize mailbox and calendar data
Enter the iPlan homepage
Query the current day's schedule
Query the current week's schedule
Query the current month's schedule
Input a keyword to view the schedule containing that keyword
Input the email sender to query related schedule
User Add Schedule
User Modify Schedule
User Delete Schedule
Query schedule by Day
User gets a schedule of the day
Query schedule by Week
User gets a schedule for the current week
Query schedule by Month
User gets a schedule for the current month
Query schedule by Keyword
User obtains a topic schedule containing the keyword
Query schedule by email sender
User obtains a schedule related to the sender of the email
Insert a new schedule, update the schedule
User gets updated schedule after adding schedule
Modify the corresponding schedule content and update the schedule
User gets updated schedule after updating schedule
Delete related schedules, update schedules
Users get updated schedules after deleting related schedules
End

1.3 Data dictionary

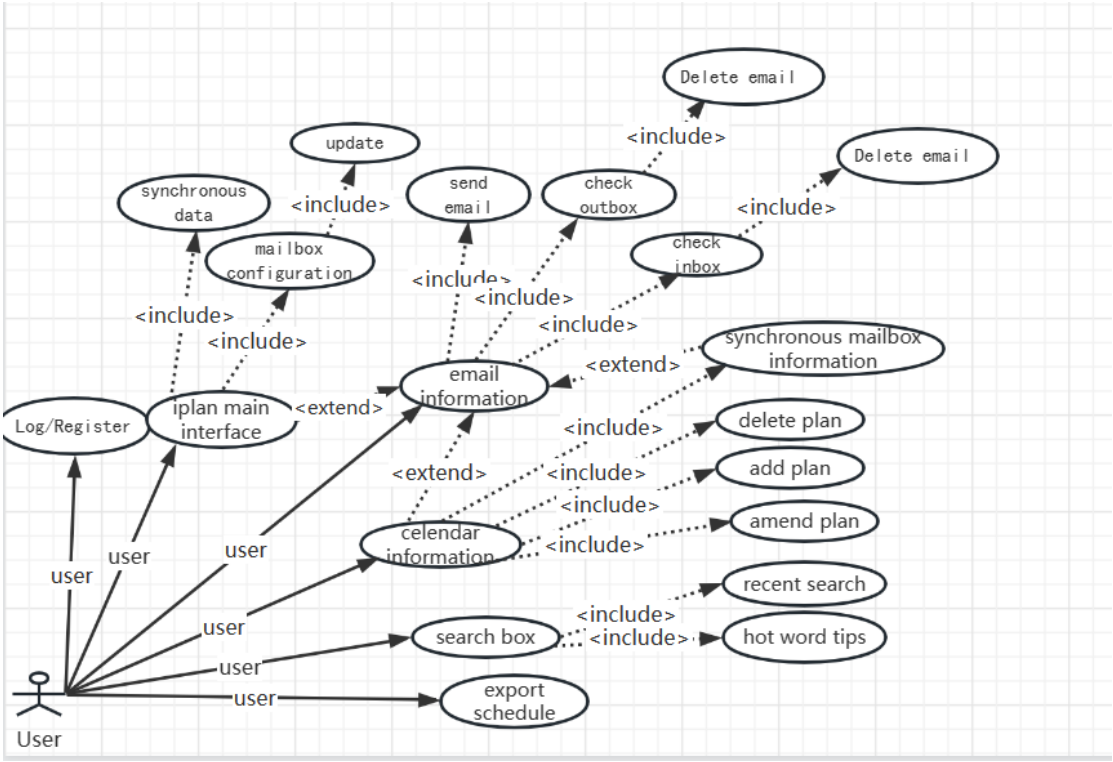
field name	data type	meaning	constraint
user_name	varchar	User login name	Non repeatable, changeable
passwords	varchar	User login password	Number+letter combination
enabled	tinyint	User login permissions	1 allows login,0does not allow login
u_id	varchar	User ID	Non repeatable
email_id	bigint	Email configuration ID	
email	varchar	Email address	Outlook email only
password	varchar	Email login password	
department	varchar	Department	
station	varchar	Post	
start_time	timestamp	Email synchronization start time	
key_word_s	varchar	Filter keywords/role tags	Interval with comma ","
key_word_t	varchar	Filter keywords/generic tags	Interval with comma ","
key_email	varchar	Filter email	Interval with comma ","
encrpyt	char	Password encryption method	1:Clear text,2:*
create_time	timestamp	Create time	
update_time	timestamp	Update time	
new_start_time	timestamp	Latest Start Time	
conference_id	bigint	Conference ID	
calender_id	varchar	Reference ID for each conference	
sender	varchar	Sender	
receiver	varchar	Receiver	
receive_time	timestamp	Receiving time	
title	varchar	Theme	

content	longtext	Content	
position	varchar	Conference venue	
start_time	timestamp	Conference start time	
end_time	timestamp	Conference end time	
create_time	timestamp	Creation time	
type	varchar	Data type	Regular email/conference
plan_id	bigint	Schedule ID	
source	char	Data sources	0:Manual add,1: Mail, 2: Meeting
id	int	Noun ID	
words	varchar	Separate nouns from the topic	
frequency	int	Frequency of noun occurrence	

1.4 Entity Relationship Diagram



1.5 Use Case Diagram



2 System Function

2.1 System Login

2.1.1 User Log On

New unregistered users can enter the user name and password in the system interface for registration. The user name cannot be repeated. The password is encrypted using the MD5 encryption algorithm, and a unique user ID will be generated after registration.

fieldName	meaning	example	constraint
user_name	User login name	mike	Non repeatable, changeable
password	User login password	Abc12345	Number and letters

u_id	User ID	9618463467	Non repeatable

2.1.2 Users Login

Registered users enter the user name and password and log in directly to the main interface.

fieldName	meaning	example	constraint
user_name	User login name	mike	Non repeatable, changeable
password	User login password	Abc12345	Number and letters

2.2 Mailbox configuration

After logging in the system, users can bind the outlook mailbox that needs to synchronize data in the first interface, and provide the address and password information to save the email password after the MD5 encryption algorithm. In addition, when binding mailbox can be set for email and calendar title filtering keywords, keywords are divided into two role tags and general tags, role tag is the user according to their need to set the custom character, general label is provided by the system, after adding filtering keywords, the system will not synchronize the keyword data in the title.

At the same time, users can add a filter mailbox, which puts the corresponding mailbox user on the blacklist, and the email sent by the account will not be synchronized.

When binding the mailbox, users need to set the start time of mailbox synchronization. If the time is set one month ago today, the system will synchronize the data from one month ago to now, and the previous data will not be synchronized. The user user is the current logged in user, and the system groups the mailbox bound by the current user name.

fieldName	meaning	example	constraint
user_name	Owning user	Alice	
email	Email address	shmtu@outlook.com	Outlook email only
password	Email login password	Abc12345	
department	Department	product department	
station	station	PM	
start_time	Email synchronization start time	yyyy-mm-dd hh:mm:ss	Timestamp
key_word_s	Filter keywords/role tags	special sale	Interval with comma ","
key_word_t	Filter keywords/generic tags	special sale	Interval with comma ","
key_email	Filter email	123456789@qq.com	

2.3 Synchronize The Mailbox and Calendar Data

2.3.1 Data Synchronization

This system uses the exchange protocol provided by Microsoft to synchronize the mail and calendar information of outlook mailbox, including the subject, content, sender and recipient; the initiator, subject, details, start and end time of the meeting of the meeting in the calendar. In the synchronization process, the data will be filtered according to the set filter keywords and blacklist mailbox. The schedule theme of the current theme is more than 80% similar to the user, and the data will not be synchronized. The synchronous content is stored in a flow data table, and the final data further integrates the calendar data displayed on the interface. In the process of synchronizing data, the system will process the email or calendar topic according to the nature of words, count the historical occurrence frequency of the corresponding vocabulary, and produce the current topic hot words. When the user binds the mailbox, the system will synchronize the data once by default, and synchronize the data regularly every day. Of course, the user can also manually synchronize it. After each synchronization of data, the start time of mailbox synchronization will be updated to

the latest receive time, as the start time of the next synchronization message, to reduce the burden of the system and avoid repeated data pulling.

2.3.2 Subject-matter Similarity Analysis

All synchronized mail and calendar topics are stored in the database. When the latest message or calendar is synchronized, the system performs similarity analysis between new topics and stored themes. If there is more than 80% similarity to a subject in the library, the message or calendar will not be stored.

2.3.3 Subject Word Segmentation And Word Frequency Statistics

The subject of email and calendar will be divided according to the nature of words, and the occurrence times of the divided words will be counted. The occurrence frequency will be accumulated, so as to obtain the 10 popular words with the highest frequency among all the topics with synchronized data.

2.4 Add or Update The Schedule

In addition to the personal schedule, the user can also be created or modified itself.

fieldName	meaning	example	constraint
user_name	Owning user	mike	
sender	Sender	shmtu@outlook.com	
receiver	Receiver	shmtu@outlook.com	
title	Theme	The first group meeting	
content	Content	task allocation	
position	Conference venue	315 conference room	
start_time	Conference start time	yyyy-MM-dd HH:mm	
end_time	Conference end time	yyyy-MM-dd HH:mm	
plan_id	Schedule ID	1001	

2.5 Delete The Schedule

The user checks the schedule to be deleted, and the background removes the corresponding schedule according to the user and schedule ID.

2.6 View The Schedule

2.6.1 To-do Schedule Today

Today's to-do schedule shows the schedule of the day, each schedule sorted by the time list. After successful login, users can enter the interface, which serves as the main interface of the program, which is convenient for users to check the unfinished schedule of today in time.

2.6.2 More Schedule

In the more schedule interface, users can choose to view all schedule information for the week of the day or all schedule information for the month of the day. In addition, the interface provides the function of searching according to the theme keywords. The user can input the keyword to query all the schedules containing the keyword in the theme. The system provides the most popular ten keywords for the user's reference. Of course, users can also query the schedule information from the sender through the sender's email address.

fieldName	meaning	example	constraint
user_name	Owning user	mike	
sender	Sender	shmtu@outlook.com	
receiver	Receiver	shmtu@outlook.com	
title	Theme	The first group meeting	
content	Content	task allocation	
position	Conference venue	315 conference room	

start_time	Conference start time	yyyy-MM-dd HH:mm	
end_time	Conference end time	yyyy-MM-dd HH:mm	
plan_id	Schedule ID	1001	

2.6.3 Schedule Export

For the schedule information, the system provides the function of exporting Excel documents, and the exported data is consistent with the data in the query interface.

fieldName	meaning	example	constraint
user_name	Owning user	mike	
sender	Sender	shmtu@outlook.com	
receiver	Receiver	shmtu@outlook.com	
title	Theme	The first group meeting	
content	Content	task allocation	
position	Conference venue	315 conference room	
start_time	Conference start time	yyyy-MM-dd HH:mm	
end_time	Conference end time	yyyy-MM-dd HH:mm	
plan_id	Schedule ID	1001	

3 System design

3.1Technology architecture

Front-end

Vue:

Vue JS, commonly referred to as Vue, is an open-source, progressive JavaScript framework created by Evan You in 2014 as an alternative to heavier frameworks like AngularJS and React. Vue combines Angular-influenced approaches and streamlined features for front-end interfacing and application development. Vue's core library is

focused on the view layer only and designed to be adopted into projects incrementally.

Vue is a JavaScript framework that facilitates the UI (user interface) development of websites and single-page applications.

A progressive JavaScript framework, Vue makes creating user interfaces simpler and more enjoyable. It was designed to be incrementally adoptable and avoid many of the baked-in fallacies of existing, monolithic frameworks.

Vue's core library is easily integrated with other libraries and existing projects. It's easy to pick up, largely due to the framework's focus on the library's view layer only. Vue is also capable of powering intricate, single-page applications through the use of modern tooling and support libraries for added functionality.

Back-end

Springboot:

Spring Boot is a microservice-based framework and making a production-ready application in it takes very less time. Spring Boot is built on the top of the conventional spring framework. So, it provides all the features of spring and is yet easier to use than spring.

- It allows to avoid heavy configuration of XML which is present in spring
- It provides easy maintenance and creation of REST end points
- Deployment is very easy, war and jar file can be easily deployed in the tomcat server
- Microservice Based Architecture

MyBatis-plus:

MyBatis is a first class persistence framework with support for custom SQL, stored procedures and advanced mappings. MyBatis eliminates almost all of the JDBC code and manual setting of parameters and retrieval of results. MyBatis can use simple XML or Annotations for configuration and map primitives, Map interfaces and Java POJOs (Plain Old Java Objects) to database records.

And MyBatis-Plus is an powerful enhanced tool for MyBatis. it provides many efficient operations for MyBatis. and you can seamlessly switch to MyBatis-Plus from MyBatis.

Mysql:

Mysql is a relational database management system. Mysql is fast, reliable, scalable, and easy to use. It was originally developed to handle large databases quickly and has been used in highly demanding production environments for many years.

Although Mysql is under constant development, it offers a rich and useful set of functions. Mysql's connectivity, speed, and security make it highly suited for accessing databases on the internet.

Mysql's key benefits include: Ease of use,Reliability,Scalability,Performance,High availability,Security,Flexibility.

EWS:

Exchange Web Services (EWS) is a cross-platform API that enables applications to access mailbox items such as email messages, meetings, and contacts from Exchange Online, Exchange Online as part of Office 365, or on-premises versions of Exchange starting with Exchange Server 2007. EWS applications can access mailbox items locally or remotely by sending a request in a SOAP-based XML message. The SOAP message is embedded in an HTTP message when sent between the application and the server, which means that as long as your application can post XML through HTTP, it can use EWS to access Exchange.

Similarity:

similarity is a Java version of the similarity toolkit that is composed of algorithms to disseminate similarity measures in natural language processing. similarity has the characteristics of practical tools, efficient performance, clear architecture, up-to-date corpus, and customizable.

Word2vec:

Word Embedding is a language modeling technique used for mapping words to vectors of real numbers. It represents words or phrases in vector space with several dimensions. Word embeddings can be generated using various methods like neural networks, co-occurrence matrix, probabilistic models, etc. Word2Vec consists of models for generating word embedding. These models are shallow two-layer neural networks having one input layer, one hidden layer, and one output layer. Word2Vec

utilizes two architectures :

CBOW (Continuous Bag of Words): CBOW model predicts the current word

Skip Gram : Skip gram predicts the surrounding context words

Our present system uses the java version of Word2vec.

HanLP Tokenizer:

In text classification (non-sentiment classification), we often only keep content words (noun, verb, form) and other words. In order to facilitate the word segmentation of text classification, HanLP provides a content word segmentation class NotionalTokenizer. At the same time, NotionalTokenizer is used by default when loading and processing the classification dataset.

EasyExcel:

EasyExcel is an Excel processing tool based on Java, which is fast, concise and can solve the problem of large file memory overflow.

It can let you do not have to consider the performance, memory and other factors under the circumstances, quickly complete Excel read, write and other functions

3.2 Datebase design

User login table: users

FieldName	Type	Length	If null	key	explain
email_id	bigint	0	NO	YES	Email configuration ID
user_name	varchar	255	NO		Owning user
email	varchar	255	NO		Email address
password	varchar	255	NO		Email login password
department	varchar	255	YES		Department
station	varchar	255	YES		Station
start_time	timestamp	0	YES		Email synchronization start time
key_word_s	varchar	4000	YES		Filter keywords/role tags
key_word_t	varchar	4000	YES		Filter keywords/generic tags
key_email	varchar	4000	YES		Filter email
encrypt	char	1	YES		Password encryption method,1.Clear text,2.*
create_time	timestamp	0	NO		Create time
update_time	timestamp	0	YES		Update time
new_start_time	timestamp	0	YES		New Email synchronization start time

Email configuration table: email_config

FieldName	Type	Length	If null	key	explain
email_id	bigint	0	NO	YES	Email configuration ID
user_name	varchar	255	NO		Owning user
email	varchar	255	NO		Email address
password	varchar	255	NO		Email login password
department	varchar	255	YES		Department
station	varchar	255	YES		Station
start_time	timestamp	0	YES		Email synchronization start time
key_word_s	varchar	4000	YES		Filter keywords/role tags
key_word_t	varchar	4000	YES		Filter keywords/generic tags
key_email	varchar	4000	YES		Filter email
encrypt	char	1	YES		Password encryption method,1.Clear text,2.*
create_time	timestamp	0	NO		Create time
update_time	timestamp	0	YES		Update time
new_start_time	timestamp	0	YES		New Email synchronization start time

Conference data table: conference_data

FieldName	Type	Length	If null	key	explain
conference_id	bigint	0	NO	YES	Conference ID
calendar_id	varchar	255	NO		Reference ID for each conference
sender	varchar	255	NO		Sender
receiver	varchar	4000	NO		Receiver
receive_time	timestamp	0	YES		Receiving time
title	varchar	500	NO		Theme
content	longtext	0	YES		Content
position	varchar	255	YES		Conference venue
start_time	timestamp	0	YES		Conference start time
end_time	timestamp	0	YES		Conference end time
create_time	timestamp	0	NO		Create time
type	varchar	50	NO		Data type：email:Message/;meeting:MeetingReques
user_name	varchar	255	YES		Owning user

Schedule data table: plan_data

FieldName	Type	Length	If null	key	explain
plan_id	bigint	0	NO	YES	Schedule ID
user_name	varchar	255	NO	NO	Owning user
sender	varchar	255	YES	NO	Sender
receiver	varchar	4000	YES	NO	Receiver
title	varchar	255	NO	NO	Theme
content	longtext	0	YES	NO	Content
position	varchar	255	YES	NO	Conference venue
start_time	timestamp	0	YES	NO	Conference start time
end_time	timestamp	0	YES	NO	Conference end time
receive_time	timestamp	0	YES	NO	Receiving time
create_time	timestamp	0	NO	NO	Create time
update_time	timestamp	0	YES	NO	Update time
source	char	1	NO	NO	Data sources 0:Manual add,1: Mail, 2: Meeting

Schedule topic noun frequency statistics table: title_frequency

FieldName	Type	Length	If null	key	explain
id	int	0	NO	YES	Noun ID
words	varchar	225	NO	NO	Separate nouns from the topic
frequency	int	0	NO	NO	Frequency of noun occurrence

3.2 Interface documentation

(1) User registration

Request method and address: POST /user/signUp

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
UserName	String	yes	user name
passWord	string	yes	login password

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information

(2) User Login

Request method and address: POST /user/signIn

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
userName	String	yes	user name
passWord	string	yes	login password

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information

(3) Obtain email configuration information

Request method and address: POST /emailConfig/getEmailConfig

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
userName	String	yes	user name

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
createTime	Timestamp	yes	creation time
department	String	no	department
email	String	yes	email address
emailId	int	yes	email ID
encrypt	int	yes	encryption method
keyWordS	String	no	Role Label
keyWordT	String	no	general label
password	String	yes	email password
startTime	String	yes	email synchronization start time
station	String	no	post
updateTime	Timestamp	yes	update time
userName	String	yes	user name

(4) Configure email information

Request method and address: POST /emailConfig/saveOrUpdate

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
department	String	no	department
email	string	yes	email address
keyEmail	string	no	filter email
keyWordS	string	no	Role Label
keyWordT	string	no	general label

password	string	yes	email password
startTime	string	yes	email synchronization start time
station	string	no	post
userName	string	yes	user name

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information

(5) Synchronize email data

Request method and address: GET /conferenceData/transferEmail

Return parameters:

Parameter name	Type	Necessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information

(6) Synchronize calendar data

Request method and address: GET /conferenceData/transferConference

Return parameters:

Parameter name	Type	Necessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information

(7) Delete Schedule

Request method and address: POST /dailyPlan/delete

Request parameters:

Parameter name	Type	Necessary to fill in	Explain
planId	String	yes	schedule id
userName	string	yes	user name

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information

(8) Query the schedule based on keywords

Request method and address: POST /dailyPlan/getByHotWords

Request parameters:

Parameter name	Type	N necessary to fill in	Explain
hotWords	String	yes	separate nouns from the topic

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
content	String	yes	content
createTime	Timestamp	yes	creation time
endTime	String	yes	conference end time
planId	String	yes	schedule id
position	String	yes	conference location
receiveTime	String	yes	receiving time
receiver	String	yes	addressee
sender	String	yes	sender
source	String	yes	data source (manual add/email/conference)
startTime	String	yes	conference start time
time	String	yes	query time
title	String	yes	theme

updateTime	Timestamp	yes	update time
userName	String	yes	user name
week	String	yes	data of the current week
sum	String	yes	Total number of query schedules

(9) Query the current day's schedule

Request method and address: POST /dailyPlan/getPlanDataByTime

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
time	String	yes	date of the day
userName	String	yes	user name

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
content	String	yes	content
createTime	Timestamp	yes	creation time
endTime	String	yes	conference end time
planId	String	yes	schedule id
position	String	yes	conference location
receiveTime	String	yes	receiving time
receiver	String	yes	addressee
sender	String	yes	sender
source	String	yes	data source (manual add/email/conference
startTime	String	yes	conference start time

time	String	yes	query time
title	String	yes	theme
updateTime	Timestamp	yes	update time
userName	String	yes	user name
week	String	yes	data of the current week
sum	String	yes	Total number of query schedules

(10) Query the current week's schedule

Request method and address: POST /dailyPlan/listByMonthRange

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
userName	String	yes	user name

Return parameters:

Parameter name	Type	Nnecessary to fill in	explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
content	String	yes	content
createTime	Timestamp	yes	creation time
endTime	String	yes	conference end time
planId	String	yes	schedule id
position	String	yes	conference location
receiveTime	String	yes	receiving time
receiver	String	yes	addressee
sender	String	yes	sender
source	String	yes	data source (manual add/email/conference

startTime	String	yes	conference start time
time	String	yes	query time
title	String	yes	theme
updateTime	Timestamp	yes	update time
userName	String	yes	user name
week	String	yes	data of the current week
sum	String	yes	Total number of query schedules

(11) Query the current month's schedule

Request method and address: POST /dailyPlan/listByTimeRange

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
userName	String	yes	user name
pageIndex	String	yes	current page number value

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
content	String	yes	content
createTime	Timestamp	yes	creation time
endTime	String	yes	conference end time
planId	String	yes	schedule id
position	String	yes	conference location
receiveTime	String	yes	receiving time
receiver	String	yes	receiver
sender	String	yes	sender

source	String	yes	data source(manual add/email/conference
startTime	String	yes	conference start time
time	String	yes	query time
title	String	yes	theme
updateTime	Timestamp	yes	update time
userName	String	yes	user name
week	String	yes	data of the current week
sum	String	yes	Total number of query schedules

(12) Add or modify the schedule

Request method and address: POST /dailyPlan/saveOrUpdate

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
content	String	yes	content
endTime	String	yes	conference end time
planId	String	yes	schedule id
position	String	yes	conference location
receiver	String	yes	receiver
sender	String	yes	sender
startTime	String	yes	conference start time
title	String	yes	theme
userName	String	yes	user name

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information

(13) Get the top ten hot words

Request method and address: GET /dailyPlanTitle/getHotWord

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
frequency	String	yes	Frequency of occurrence
id	String	Yes	Noun id
words	String	yes	Separate nouns from the topic

(14) Query the schedule based on email

Request method and address: POST /dailyPlan/getFromEmail

Request parameters:

Parameter name	Type	Nnecessary to fill in	Explain
email	String	yes	Email address

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
content	String	yes	content
createTime	Timestamp	yes	creation time
endTime	String	yes	conference end time
planId	String	yes	schedule id
position	String	yes	conference location
receiveTime	String	yes	receiving time

receiver	String	yes	addressee
sender	String	yes	sender
source	String	yes	data source (manual add/email/conference)
startTime	String	yes	conference start time
time	String	yes	query time
title	String	yes	theme
updateTime	Timestamp	yes	update time
userName	String	yes	user name
week	String	yes	data of the current week
sum	String	yes	Total number of query schedules

(15) export the schedule

Request method and address: POST /dailyPlan/exportExcel

Request parameters:

Consistent with the current query interface request parameters.

Return parameters:

Parameter name	Type	Nnecessary to fill in	Explain
code	String	yes	Interface status code
message	String	yes	Interface information
data	Object	yes	data set

data：

Parameter name	Type	Nnecessary to fill in	Explain
content	String	yes	content
createTime	Timestamp	yes	creation time
endTime	String	yes	conference end time
planId	String	yes	schedule id
position	String	yes	conference location
receiveTime	String	yes	receiving time

receiver	String	yes	addressee
sender	String	yes	sender
source	String	yes	data source (manual add/email/conference)
startTime	String	yes	conference start time
time	String	yes	query time
title	String	yes	theme
updateTime	Timestamp	yes	update time
userName	String	yes	user name
week	String	yes	data of the current week
sum	String	yes	Total number of query schedules

4. requirement items and task allocation

Our project is front-end and back-end separated，the total number of required items is 33. Among them, the database design and interface document design have been completed. The remaining 31 items will be completed in four time nodes.

requirement items：

requirement code	requirement content	milestone	developer
FD20231001	user registration front-end developmen	2023/04/20	Huang Deyu
BD20231002	user registration back-end developmen	2023/04/20	Huang Deyu
FD20231003	user login front-end developmen	2023/04/20	Huang Deyu
BD20231004	user login back-end developmen	2023/04/20	Huang Deyu
FD20231005	configure email information front-end developmen	2023/05/04	Huang Deyu
BD20231006	configure email information back-end developmen	2023/05/04	Huang Deyu
FD20231007	synchronize email data front-end developmen	2023/05/18	Liu Zhiqing
BD20231008	synchronize email data back-end developmen	2023/05/18	Xie Guangwei

FD20231009	synchronize calendar data front-end developmen	2023/04/20	Liu Zhiqing
BD20231010	synchronize calendar data back-end developmen	2023/05/18	Xie Guangwei
BD20231011	divide topics into parts of speech back-end developmen	2023/05/18	Xie Guangwei
BD20231012	count title words frequency back-end developmen	2023/05/18	Xie Guangwei
BD20231013	analyze title words similarity back-end developmen	2023/05/18	Huang Deyu
FD20231014	add schedule front-end developmen	2023/04/20	Liu Zhiqing
BD20231015	add schedule back-end developmen	2023/04/20	Huang Deyu
FD20231016	update schedule front-end developmen	2023/04/20	Liu Zhiqing
BD20231017	update schedule back-end developmen	2023/04/20	Huang Deyu
FD20231018	delete schedule front-end developmen	2023/05/04	Liu Zhiqing
BD20231019	delete schedule back-end developmen	2023/05/04	Huang Deyu
FD20231020	search the day’s schedule front-end developmen	2023/04/20	Liu Zhiqing
BD20231021	search the day’s schedule back-end developmen	2023/04/20	Xie Guangwei
FD20231022	search the week’s schedule front-end developmen	2023/05/04	Liu Zhiqing
BD20231023	search the week’s schedule back-end development	2023/05/04	Xie Guangwei
FD20231024	search the month’s schedule front-end developmen	2023/05/18	Liu Zhiqing
BD20231025	search the month’s schedule back-end development	2023/05/18	Xie Guangwei
FD20231026	search schedule by keyword front-end development	2023/05/31	Liu Zhiqing
BD20231027	search schedule by email back-end development	2023/05/31	Huang Deyu
FD20231028	search calendar by email front-end development	2023/05/31	Liu Zhiqing
BD20231029	search schedule by email back-end development	2023/05/31	Huang Deyu
FD20231030	export schedule Excel front-end development	2023/05/31	Liu Zhiqing

BD20231031	export schedule Excel back-end development	2023/05/31	Xie Guangwei
ID20231032	interface documentation desgin	finished	Huang Deyu
DB20231033	database design	finished	group

Milestone

Finished

ID20231032、DB20231033

2023.04.20 week 10

FD20231001、BD20231002、FD20231003、BD20231004、FD20231009、FD20231014

BD20231015、FD20231016、BD20231017、FD20231020、BD20231021

2023.05.04 week 12

FD20231005、BD20231006、FD20231018、BD20231019、FD20231022、BD20231023

2023.05.18 week 14

FD20231007 、 BD20231008 、 BD20231010 、 BD20231011 、 BD20231012 、 BD20231013

FD20231024、BD20231025

2023.05.31 week 16

FD20231026、BD20231027、FD20231028、BD20231029、FD20231030、BD20231031

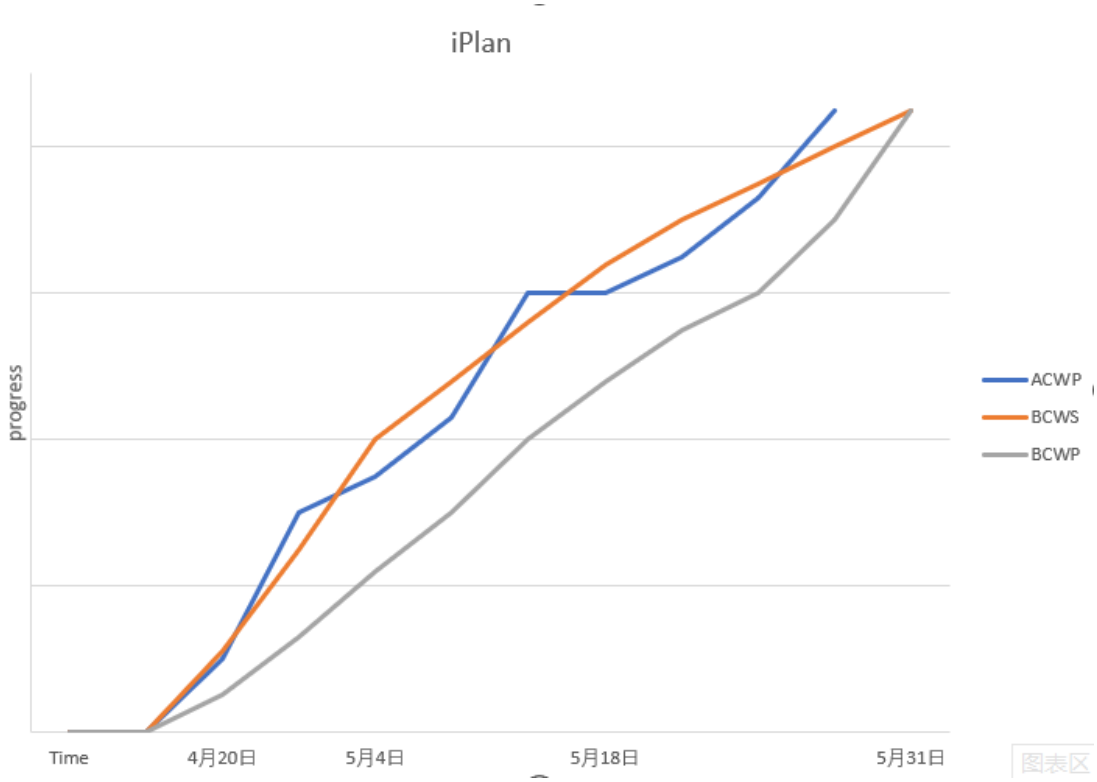
5. Risk Management Plan

In software project management, risk control is a very important link, the purpose of risk control is to reduce the impact of risk as much as possible, to ensure that the project can be completed on time, according to the quality, according to the budget. The following table is a detailed description of risk control in software project management:

Risk Category	Risk Incidence	Impact Level	Risk Description	Risk Management Plan	Practical Measures

Front-end Interface Design	**high**	**high**	As team members have not systematically learned front-end technology, core functions may occur and cannot be implemented.	regularly evaluate progress and find solutions to ensure that the project can proceed as planned.	Strengthen the front-end knowledge level of learning，andSeek the help of someone online or in the real world who understands this technique.
Framework Design	**high**	**high**	The development team may encounter technical difficulties due to the use of new technology.For example, the vue3 technology in our Iplan project.	Establish a technical group including external team members to be able to solve problems quickly when they arise.	Regularly schedule technical reviews to ensure the project is on schedule, and adjust technical solutions if necessary.
Software Functionality Development	**medium**	**high**	Continuous changes in requirements may result in project delays or exceed budgets,Even induce team members to complain and not communicate	Implement a clear requirement change process, clarify the person responsible for the change, and ensure that each change is sufficiently approved and evaluated.	Discuss project changes with team members, incorporate necessary changes into the project management plan, and fully communicate and coordinate to avoid unnecessary delays and extra time costs.
Software Functionality Testing	**low**	**low**	The project may face financial problems due to insufficient funds or budget overruns.	Regularly monitor the budget and seek ways to reduce costs.	Regularly conduct budget reviews and take corrective action when necessary.
Unforeseen Circumstances	**low**	**high**	Epidemic or team member may temporarily leave the project due to health reasons.	Establish a multi-level team cross-training mechanism to ensure that multiple team members can take over the core member's responsibilities.	Prepare backup personnel for the core member to ensure that core functions can continue when necessary.
Classroom Demonstrations	**low**	**high**	if the final classroom presentation, management decisions and planning without considering the actual situation may lead to poor project presentation. This can lead to lower grades and even misjudgments about incomplete projects.	Carefully and clearly explain project planning and tasks, establish effective communication mechanism and process, ensure smooth communication between project team members, and finally present good results.	Reinforce details in class presentations; Seek support and solutions from team members in difficult situations; Adjust the methods in time to ensure the successful completion of project management courses.

Cost Shchedule Control:



From the above figure, we can see the trend lines of BCWS, ACWP, and BCWP in this project. BCWS represents the budgeted cost of actual work completed, BCWP represents the budgeted cost of planned work, and ACWP represents the actual cost of

completed work. Based on the above situation, the comparison of values of BCWP, BCWP, and ACWP at each checkpoint is as follows:

- April 20th checkpoint

  BCWS：

  BCWP：（Below average）

  ACWP：（Below average）

The reason for the lower ACWP is that one of the 11 items that should have been submitted in the first submission was missed, resulting in a lower ACWP value. There was a delay, and the missing item was submitted on the second day.

- April 20th to May 4th

  BCWS

  BCWP：（Below average）

  ACWP：（From low to high）

Due to the submission of an additional item, the actual cost of work (ACWP) will be slightly higher than the planned cost of actual work. Then, at the next checkpoint, there was a delay due to technical reasons, resulting in the ACWP being slightly lower than the planned cost of actual work.

- May 4th

  BCWS：

  BCWP：（Below average）

  ACWP：（Below average）

Due to the delay in task completion, both BCWP and ACWP values will deviate from the expected levels. On May 4th, the submission could not be made on time due to technical reasons. The reason was that this project is a front-end and back-end separated project. During the debugging of the front-end and back-end, data transmission could not be achieved successfully. After inspection, it was found that there were naming inconsistencies and errors in the front-end API scheduling code. After two days of debugging, the issue was resolved and the missing item was submitted.

- May 4th to May 18th

  BCWS

BCWP：（Below average）
ACWP：（From low to high）

The actual work of this project exceeded the planned budgeted cost, so BCWP is in a low state. During this period, it was necessary to solve the task of delayed submission due to technical reasons and complete the task of the next checkpoint. Therefore, ACWP changed from low to high.

- May 18th

  BCWS：

  BCWP：（Below average）

  ACWP：（Below average）

Some technical issues were resolved at the previous checkpoint, so the task completion afterwards was very smooth. The task was completed 5 days ahead of schedule, and both BCWP and ACWP values were slightly lower than the expected levels. That is, the actual cost and planned cost will be lower than the budgeted cost for the actual work completed.

- May 18th to May 31th

BCWS
BCWP：（Below average）
ACWP：（From low to high）

As the task was completed ahead of schedule at the checkpoint on May 18th, the actual cost of work (ACWP) will be slightly lower than the planned cost of actual work (BCWS). Later during this period, the task of the next checkpoint was delivered ahead of schedule, so the actual cost of work will experience a change from low to high.

- May 31th

BCWS
BCWP：
ACWP：

Finally, at the checkpoint on May 31st, all tasks were completed, and the values of BCWP, BCWS, and ACWP reached the expected levels. The project was completed.

Overall, from April 20th to May 18th, the actual cost of work (ACWP) deviated

significantly, especially at the checkpoint on April 20th where the ACWP value was high, indicating an issue with project cost control. It is necessary for the foundation to take appropriate measures to reduce costs and ensure the project is completed smoothly according to the budget and schedule.

6. Test plan and Test cases

Our project will use JUnit5 for automated testing of project functionality. JUnit is one of the most popular automated testing frameworks in Java, used for writing and running unit tests. JUnit has extensive community support and well-documented resources, as well as being easy to learn, powerful, and flexible, making it widely used in the Java development field.

The main features of JUnit include:

Test runners: JUnit provides a test runner that can run test cases in different testing environments.

Annotations: JUnit uses annotations to identify test methods and test classes. For example, the @Test annotation is used to identify test methods, while the @RunWith annotation is used to identify the test runner for running tests.

Assertions: JUnit provides multiple assertion methods for verifying whether test results meet expectations. For example, the assertEquals() method is used to compare whether two objects are equal.

Extensibility: JUnit can be extended by combining it with other frameworks, such as the Mockito framework for testing object simulation, and the Spring Test in the Spring framework to support testing Spring applications.

Plugin support: JUnit supports many plugins, such as Eclipse plugins, Ant tasks, and Maven plugins.

Furthermore, JUnit also has the following advantages:

Automatic execution of test code: JUnit provides methods that allow the test writer to execute the test code, thereby focusing on the execution conditions and

results.

Save test time: JUnit can automatically run multiple test cases, especially suitable for unit testing, which can quickly locate and solve problems.

Accurate reporting of error messages: JUnit provides detailed error reports, enabling developers to quickly locate and fix errors.

Here is a specific example of using JUnit for testing:

Json group：contains three test cases

```
[
  {
    "userName": "",
    "sender": "",
    "receiver": "",
    "title": "",
    "content": "",
    "position":"",
    "startTime": "",
    "endTime": "",
    "planId": ""
  },
  {
    "userName": "admin1",
    "sender": "lidanqing <lidanqing@vincce.com>",
    "receiver": "lidanqing <lidanqing@vincce.com>,chenbowen <chenbowen@outlook.com>,",
    "title": "测试会议99",
    "content": "全体会议",
    "position":"401会议室",
    "startTime": "2023-10-25 23:00:00",
    "endTime": "2023-10-25 24:00:00",
    "planId": ""
  },
  {
    "userName": "admin1",
    "sender": "lidanqing <lidanqing@vincce.com>",
    "receiver": "lidanqing <lidanqing@vincce.com>,chenbowen <chenbowen@outlook.com>,",
    "title": "长度测试这种颜色常用于设计和网页开发中，用于突出显示或强调某些元素，以增强视觉效果和吸引用户注意力。具体使用时需要根据具体情况来选择合适的颜色和搭配，以达到最佳的视觉效果",
    "content": "全体会议",
    "position":"401会议室",
    "startTime": "2023-10-25 23:00:00",
    "endTime": "2023-10-25 24:00:00",
    "planId": ""
  }
]
```

The first test case has empty parameters and is used to test the interface method's null pointer handling; the second case tests normal data to see if the functionality meets expectations; the third case contains long strings and tests the handling of large text.

Method URL：

```
private static final String saveOrUpdate = "/dailyPlan/saveOrUpdate";
```

Read JSON document as input parameter：

```
@Value("classpath:data/queryPlanDataSaveOrUpdate.json")
private Resource queryPlanDataSaveOrUpdate;
```

Automated testing start button：

```
@Test
public void saveOrUpdate() throws Exception{
    AddDailyPlanRequest request = objectMapper.readValue(queryPlanDataSaveOrUpdate.getInputStream(),
            AddDailyPlanRequest.class);
    AddDailyPlanResponse response = restPost(saveOrUpdate,request,AddDailyPlanResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
```

We can see the following information after executing the method:

An error will be thrown when executing the first test case.

```
com.fasterxml.jackson.databind.exc.MismatchedInputException: Cannot deserialize instance
```

Because all the input parameters are empty, it violates the constraint conditions of the request entity in the method, so it cannot be serialized correctly. This means that an empty form cannot be passed when passing values across the entire end.

The second and third test cases were executed correctly without any errors, and returned the following information:

request：

```
MockHttpServletRequest:
      HTTP Method = POST
      Request URI = /dailyPlan/saveOrUpdate
       Parameters = {}
          Headers = [Content-Type:"application/json", Accept:"application/json", Content-Length:"575"]
             Body = <no character encoding set>
    Session Attrs = {}
```

response：

```
MockHttpServletResponse:
           Status = 200
    Error message = null
          Headers = [Content-Type:"application/json;charset=utf-8"]
     Content type = application/json;charset=utf-8
             Body = {"code":0,"message":"Successfully"}
```

When the system status code returns 200 and the custom return status code is 0, it means that the method has been executed successfully. This is an insert operation, and after the method is executed, it checks whether the data has been successfully added to the database table. Finally, this confirms that the method has passed the test.

Our project has a total of 12 interface methods for automated testing, with 22 test cases. The testing methods will be grouped according to different modules. All the testing methods are shown below:

ConferenceDataControllerTest：

```
private final static String transferEmail = "/conferenceData/transferEmail";
private final static String transferConference = "/conferenceData/transferConference";
@Test
public void transferEmail() throws Exception{
    TransferConferenceDataResponse response = restPost(transferEmail,"",TransferConferenceDataResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void transferConference() throws Exception{
    TransferConferenceDataResponse response = restPost(transferConference,"",TransferConferenceDataResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
```

EmailConfigControllerTest：

```
private static final String insertOrUpdateMail = "/emailConfig/saveOrUpdate";
private static final String mailSelect = "/emailConfig/getEmailConfig";
private static final String getFromEmail = "/emailConfig/getFromEmail";

@Value("classpath:data/querySaveOrUpdate.json")
private Resource querySaveOrUpdate;
@Value("classpath:data/queryGetEmailConfig.json")
private Resource queryGetEmailConfig;
@Value("classpath:data/queryGetFromEmail.json")
private Resource queryGetFromEmail;

@Test
public void insertOrUpdateMail() throws Exception{
    AddEmailConfigRequest request = objectMapper.readValue(querySaveOrUpdate.getInputStream(),AddEmailConfigRequest.class);
    AddEmailConfigResponse response = restPost(insertOrUpdateMail,request,AddEmailConfigResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void mailSelect() throws Exception{
    SelectEmailConfigRequest request = objectMapper.readValue(queryGetEmailConfig.getInputStream(),SelectEmailConfigRequest.class);
    SelectEmailConfigResponse response = restPost(mailSelect,request,SelectEmailConfigResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void getFromEmail() throws Exception{
    SelectConferenceDataByToEmailRequest request = objectMapper.readValue(queryGetFromEmail.getInputStream(),
    SelectConferenceDataByToEmailRequest.class);
    SelectFromEmailResponse response = restPost(getFromEmail,request,SelectFromEmailResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
```

PlanDataControllerTest：

```
private static final String selectByTimeRange = "/dailyPlan/listByTimeRange";
private static final String selectPlanDataByTime = "/dailyPlan/getPlanDataByTime";
private static final String selectByMonthRange = "/dailyPlan/listByMonthRange";
private static final String saveOrUpdate = "/dailyPlan/saveOrUpdate";
private static final String dailyPlanDeleteById = "/dailyPlan/delete";
private static final String selectByHotWords = "/dailyPlan/getByHotWords";

@Value("classpath:data/querySelectByTimeRange.json")
private Resource querySelectByTimeRange;
@Value("classpath:data/querySelectPlanDataByTime.json")
private Resource querySelectPlanDataByTime;
@Value("classpath:data/querySelectByMonthRange.json")
private Resource querySelectByMonthRange;
@Value("classpath:data/queryPlanDataSaveOrUpdate.json")
private Resource queryPlanDataSaveOrUpdate;
@Value("classpath:data/queryDailyPlanDeleteById.json")
private Resource queryDailyPlanDeleteById;
@Value("classpath:data/querySelectByHotWords.json")
private Resource querySelectByHotWords;
```

```
@Test
public void selectByTimeRange() throws Exception{
    SelectDailyPlanByTimeRangeRequest request = objectMapper.readValue(querySelectByTimeRange.getInputStream(),
            SelectDailyPlanByTimeRangeRequest.class);
    SelectTimeAxisPlanDataResponse response = restPost(selectByTimeRange,request,SelectTimeAxisPlanDataResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void selectPlanDataByTime() throws Exception{
    SelectPlanDataByTimeRequest request = objectMapper.readValue(querySelectPlanDataByTime.getInputStream(),
            SelectPlanDataByTimeRequest.class);
    SelectPlanDataResponse response = restPost(selectPlanDataByTime,request,SelectPlanDataResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void selectByMonthRange() throws Exception{
    SelectDailyPlanByMonthRangeRequest request = objectMapper.readValue(querySelectByMonthRange.getInputStream(),
            SelectDailyPlanByMonthRangeRequest.class);
    SelectPlanDataResponse response = restPost(selectByMonthRange,request,SelectPlanDataResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void saveOrUpdate() throws Exception{
    AddDailyPlanRequest request = objectMapper.readValue(queryPlanDataSaveOrUpdate.getInputStream(),
            AddDailyPlanRequest.class);
    AddDailyPlanResponse response = restPost(saveOrUpdate,request,AddDailyPlanResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void dailyPlanDeleteById() throws Exception{
    DeleteDailyPlanRequest request = objectMapper.readValue(queryDailyPlanDeleteById.getInputStream(),
            DeleteDailyPlanRequest.class);
    DeleteDailyPlanReponse response = restPost(dailyPlanDeleteById,request,DeleteDailyPlanReponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
@Test
public void selectByHotWords() throws Exception{
    SelectPlanDataByHotWordRequest request = objectMapper.readValue(querySelectByHotWords.getInputStream(),
            SelectPlanDataByHotWordRequest.class);
    SelectPlanDataByHotWordsResponse response = restPost(selectByHotWords,request,SelectPlanDataByHotWordsResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
```

TitleFrequencyControllerTest：

```
private static final String listHotWord = "/dailyPlanTitle/getHotWord";

@Test
public void listHotWord() throws Exception{
    SelectHotWordsResponse response = restPost(listHotWord,"",SelectHotWordsResponse.class);
    assertThat(response.getCode(),equalTo(ErrorCode.SUCCESS.getCode()));
}
```

Because there are too many test cases, we will provide the code instead. All the cases are designed based on null pointers, long strings, special formatting characters, and other aspects, similar to the above example, but adjusted based on the characteristics of each method. Please refer to the source code in the project repository for details.