Group Report for Software Engineering Group Project

2021/2022 Semester 2
<Project Title> Leisure Town
<Date> 22th April, 2022

Group Members:

1. <ruidong.shen, 1928071>
2. <sirui.li, 1931020>
3. <xinyi.huang, 1927134>
4. <junhe.yan, 1929102>
5. <ziyu.xiong, 1929542>
6. <chuanzhi.su, 1931018>
7. <ruowei.xing, 1927922>
8. <lincheng.shi, 1927978>
9. <zhaoyu.xu, 1930717>

Catalogue

Introduction

Our web app is called Leisure Town. It's a platform to share life and find friends with same interests. This report and a single zip file will be uploaded in learningmall. The zip file includes front-end project (folder "blog-app"), back-end project (folder "Leisure_Town"), database (file "new_dataset.sql") and redis (folder "redis1" for Windows, folder "redis2" for MacOS). Considering the legal, social, ethical and professional issues, we used JWT (JSON Web Token) technology when logging in, and put the token into redis as double insurance. When logging in for authentication, it will first verify whether the token character is legal, and then go to redis to verify whether it exists. This greatly improves security.

Problem Statement

Nowadays, we are in the information age with highly developed network, people's work efficiency is getting higher and higher, and the pace of life is getting faster and faster. As a result, people nearly cannot have spare time and the pressure and anxiety of the younger generation are also increasing.
However, there is no platform on the Internet for them to share their lives and relax freely. The main social media that people are using now has a common issue — They focus on business goals too much. To be specific, currently, people use social app to add many colleagues and customers for easy communication. Meanwhile, people always need to post work-related content rather than their lives.

Hence, they need a platform that only to share their own ideas and making friends with the same interests.

They would like to post the articles at any time and anywhere, upload photos to the platform as their album, type blog posts or some notes on the platform and use it as a notebook, and be able to view relevant blogs based on their interests.

The following illustrate several decriptions of the problems to be addressed and improved upon for most social platform:

- User cannot share more blogs on lives other than business goals and work
- Users cannot publish blogs due to the too short word limit. .
- User face a too complex interface and has a lot of functionality that they don't normally use and hard to operate.
- Users must add friends first to view or comment on other people's blogs.

- Users cannot post blogs according to categories.
- Users cannot post blogs with many tags.
- Users cannot see which tags of blog are the most popular.
- Users cannot view the blogs specifically based on their preferable categories and tags.
- Users cannot choose a few function, such as changing font size and styles or immersive reading when writing blogs.
- Users cannot know how many times their blog has been viewed.
- Users cannot have a rough idea of blog content since blogs do not have an abstract for being shown in square.

Aims of The Project

Based on the problems listed above, our social platform hope to achieve several goals:

1. Users share their lives and ideas for relaxing other than purposes of work
2. Blogs content tend to become more vivid compared to current social medias
3. Let users make friends who are interested in blogs on the same topic
4. Let users share their lives and ideas with everybody rather than the ones restricted in a friend circle

Make a social platform that easy to operate without any complex interface and function

- The product owner wants Leisure Town online after 10 weeks. To achieve the requirement, we plan to finish the project in 3 phases.
- Implementation of register and login system
- Implementation of the main page, posting and blog tags
- Implementation of trending tags, comments, like and view count

Project Scope

Range of product:

The main goal of this project is to develop a web App and make it have a user system. Users can share some posts on this platform. The product also has a number of minor additional features that complement these major functions. In the final deliverable, the product shall have a series of functions such as registration, login, data verification from the background, page jump, article label, article classification, view article details, publish article, editor, comment and so on.

Scope of project:

In order for these functions to be possible, the product should have a database. In the registration and login part shall receive data from the interface provided by the back end and verify the data; UI page design requires the dependency of Elements. UI and other components; In order to realize the page jump, and the data request need to establish the corresponding route and interface.

Objectives

User Characteristics

Users involved in the system:

1. The people with internet access, including:
   Cultural celebrities;
   Business executives;
   Internet celebrities;
2. General public and other social groups

Characteristics of the end users that may influence UI/UX design:

a) Users who have more free time in daily life

For the users who have more free time in daily life, our platform offers more word count limit for a blog compared to current microblog media.

b) Users who post blogs with non-profit purposes

Our platform does not provides any monetary reward mechanism between bloggers and followers, aiming to shape an environment where any bloggers only intend to share their ideas and find friends in terms of specific tags and categories.

c) Users mainly with a relatively high age range which is between 25 and 50

Our website intends to use simple UI/UX design in order to adapt to the favor of relative high age range, which reflects mainly on these aspects [1] [2]:

1. Relative large font size
2. Less color matching
3. Quick start function design
4. Less modern elements

Constraints

This product is a blog platform mainly served for people from 25 to 50, and has the functions of publishing articles, classifying articles and commenting. It is a platform to help people share their lives or experiences in order to reduce the pressure in this busy era. Project constraints are divided into the following aspects.

First, constraints on project scope; Based on the objectives and functions of this project, the main audience of this project will be people from 25 to 50 under great pressure. For other age groups, the simple, clear pages and operation mode of the product may be inappropriate to a certain extent.

Second, the time constraint of the project; Due to the short development time of this project and the lack of development experience of the team, the work efficiency was limited. We developed the product while learning relevant knowledge, and only completed the development of the main functions expected. Other functions will be described in the future plan.

Finally, the cost constraint of the project; The cost is a development team of nine people and their own equipment.

Assumptions and Dependencies

Assumptions:

1. A good network environment and remain stable throughout the project life cycle.
2. The browser used by users must be secure, stable, fast and smooth.
3. All the physical devices such as laptops, computers, mobile phones, and more are in good condition to be used when conducting the project.

Dependencies:

-Finish-to-Start:

1. Users cannot login until they have registered.
2. Users cannot write blogs and comments on blogs until they have logged in with account.
3. Users cannot view blogs and check blogs details until blogs have been published.

4. Users cannot give comments until blogs have been published.
5. Our web app connot be started untill all the configuration and production environment are deployed correctly.

-Start-to-Start:
1. Users cannot use editor until they start to write articles.

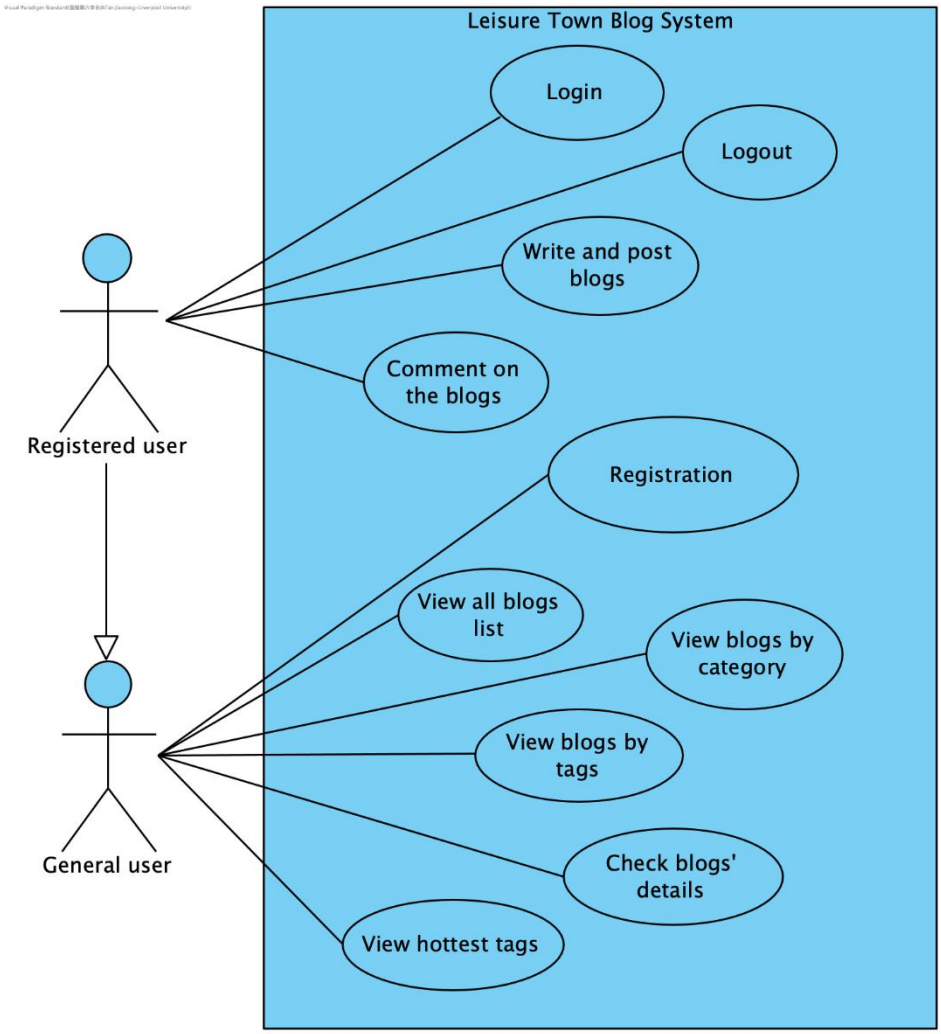
-Finish-to-Finish:
1. Blogs cannot be published until heading, content, categories and tags have been selected, abstracts heve been written.
2. "Go top" button cannot be displayed in main page and passage detail page until you have scrolled the page to some extend.
3. "Go top" button cannot be displayed in main page until the page has had 7 passages.
4. Registeration cannot be successful until the account name are checked without repetition successfully in the database.
5. Registeration cannot be successful until the format of each item have be checked without mistakes.
6. Login cannot be successful until the account name and password are checked without mistakes in the database.

Software Requirements

1. Blogs management
    - 1.1. Write and post blogs
    - 1.2. View all blogs list
    - 1.3. View blogs by categories
    - 1.4. View blogs by tags
    - 1.5. Check blogs' details
    - 1.6. Comment on the blogs
    - 1.7. Search blogs by blog content
    - 1.8. Like/Thumbs up blogs
    - 1.9. Trendy blogs
    - 1.10. View trendy tags
2. User's account management
    - 2.1. Registration
    - 2.2. Login and logout
3. User's profile management
    - 3.1. View profile
    - 3.2. Edit profile

System Design

System Overview



Use Case UC1: Login

Primary Actor: Registered user
Preconditions: The user has an account

Post conditions: Funtions of writing blogs, posting blogs and comment on the blogs can be used
Main Successful Scenario:

1. Registered user enters the website to the home page
2. Registered user click the "Login" button on the navigation bar
3. The page switches and a login box with two input boxes of user name and password is displayed
4. Registered user input user name and password
5. Registered user clicks the "Login" button and login successfully
6. The web page skips from login page to the home page
7. The original place of "Logcin" is replaced by the picture of users

Exception Scenarios:

1. If the username or passwordis incorrect,a message "username or password is not existed" is displayed, and the login cannot be successful.
2. If the user clicks login button without username, a message "Please enter the username" is displayed and the login cannot be successful.
3. If the user clicks login button without password, a message "Please enter the password" is displayed and the login cannot be successful.

Use Case UC2: Log out

Primary Actor: Registered user
Preconditions: Registered user has logged in with account
Post conditions: Registered user logs out the system
Main Successful Scenario:

1. Registered user enters the home page
2. Registered user clicks their head portrait of user on the navigation bar
3. A drop-down list with item writing "Log out" is displayed from the head portrait of user
4. Registered user clicks "Log out" item
5. Registered user logs out successfully

Use Case UC3: Write and post blogs

Primary Actor: Registered and logged in user
Preconditions: The user who has registered and logged in clicked the "Write an article" button at the top of Leisure Town's home page to write a new blog

with the editer. Finishing editing the blog, the user clicked the "Post" button to post the blog

Post conditions: A new articles with title, author, article content, article summary, article classification and article tags will be published successfully in Leisure Town

Main Successful Scenario:

1. Registered users login with their account
2. Registered users click the "Publish an article" button of Leisure Town navigation bar in main page to switch the page to write.
3. Registered user writes the title of the article and writes the content of the article, and user can use the editor to complete their blogs.
4. Registered users can according to their own requirements to beautify or design layout, including content in bold, italics, different level title, underline, marking, tag, the Angle, the Angle of standard, paragraph references, ordered list, unordered list, add images, the code block, open title navigation, preview, full-screen editor, immersive reading,markdown syntax help (can add tables, formulas, etc.), step up, step down, empty.
5. Registered users click the "publish" button, switch to a pop-up window, write a summary of the article, select the article category, check the article tags.
6. Registered users click the "publish" button, the article is published successfully and a pop-up window shows that "Published successfully".

Exception Scenarios:

1. If the title is more than 20 characters long, an error message is displayed and the publication cannot be successful.
2. If the title or article content is not written yet, directly click press conference prompt "title cannot be empty","content cannot be empty ".
3. If the article summary or article classification or tags is not filled before, directly press conference prompt "please input summary" or "please select article classification" or "please select tags", summary cannot be more than 80 characters
4. Click "Cancel publication" on the page of writing an article and a prompt box will appear. Select "Cancel" or "×" to go back to the page of write post and select "OK" to enter the main page of Leisure Town. The system will restore the article, summary, tag and categories before user cancel the publication.

Use Case UC4: Comment on the blogs

Primary Actor: Logged in user

Preconditions: The user who has registered and logged in can check blog's details
Post conditions: A new comment is added to the blog successfully, and user can see his/her comment after reloading the page
Main Successful Scenario:

1. Registered user logins with an account
2. Registered user clicks a blogs and check blog's detail
3. Registered user enters some comments in the box which below the blog's content.
4. Registered user clicks "Post" button
5. A comment is added to the blog successfully

Exception Scenarios:

1. If the user name has not login and post a comment, a message "Please login"is displayed and the comment posting cannot be successful.

Use Case UC5: Check blog's detail

Primary Actor: General user
Preconditions: The user has clicked Leisure Town's home page.
Post conditions: All of the details of this blog have been shown on the blog's page.
Main Successful Scenario:

1. General user enter website turns to the main page
2. General user clicks any article title to view the details of the blog, the page to jump to the blog page.
3. The blog page displays the title, content, date of publication, author profile picture, number of views, tags used, category of posts and comments.
4. From anywhere on the blog page, click the button on the right to return to the top of the page.

Use Case UC6: View blogs by tags

Primary Actor: General user
Preconditions: There are Blogs post with tags by users, and they can be classified according to tags. And one blog will be post with at least one tag. The user can click the "town tag" button.

Post conditions: The blogs with the same tag are put together and displayed in time order.
Main Successful Scenario:

1. General user enters the website to the home page
2. General user clicks "town tag" button which is in the navigation bar of the home page
3. All tags of blogs are displayed in the screen
4. General user selects one tag
5. All blogs with the same tag are displayed in publication time order

Use Case UC7: View hottest tags

Primary Actor: General user
Preconditions: The user has clicked Leisure Town's home page and some blogs have been published.
Post conditions: Displays the six hottest tags in the database on the main page.
Main Successful Scenario:

1. Users, both registered and unregistered, clicked on the Leisure Town home page.
2. On the right side of the home page, there is a module that displays the six most popular tags already in the database, based on the number of tags used for articles.
3. Click "View All" button to go to the Town's Tag page and view all the tags.

Use Case UC8: View blogs by category

Primary Actor: General user
Preconditions: There are tags classified by registered users. There are blogs post by registered users. There is at least one blog with only one category post by registered users.
Post conditions: The blogs with the same categories are put together and displayed in time order.
Main Successful Scenario:

1. General user enters the website to the home page
2. General user clicks "town category" button which is in the navigation bar of the home page
3. All categories of blogs are displayed in the screen
4. General user selects one category
5. All blogs with the same tag are displayed in publication time order

Use Case UC9: View all blogs list

Primary Actor: General user
Preconditions: General user and registered user enter the website. There is at least one blog post by registered users.
Main Successful Scenario:

1. General user enters the website to the home page
2. The website display all blogs lists in home page

Use Case UC10: Registration

Primary Actor: General user
Preconditions: General user has not registered before.
Post conditions: General users can login the website with account and become the registered users.
Main Successful Scenario:

1. General user enters the website to the home page
2. General user clicks the "Registration" button on the navigation bar
3. The page switches and a login box with three input boxes of user name, nick name and password is displayed
4. General user input user name, nick name and password
5. General user clicks the "Registration" button and registrate successfully
6. The web page skips from login page to the home page

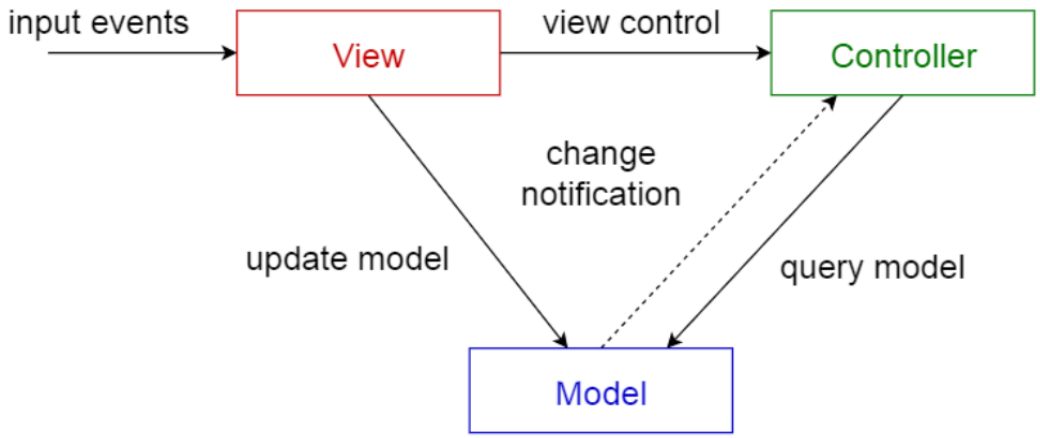
Exception scenario:

1. If the username already exists, a red message " Account already exist" is pop up on the top of the page and the registration is not successful
2. If the general users cannot input the username, nick name and password all to the input box, the red message "Please enter username""Please enter nick name" "Please enter password" will be displayed below each input box and the registration is not successful

System Architecture

In this project, our overall framework is MVC (Model-View-Controller) framework. "M" stands for Model. Models represent business rules. "V" stands for View. A View is the interface that the user sees and interacts with. "C" stands for Controller. A controller is a controller that accepts input from the user and invokes models and views to fulfill the user's requirements.

MVC is a monomer architecture as well as a back-end separation architecture. Thus, we use different architectural techniques on the front end and the back end.



The main front end we use is vue.js framework. Vue is a set of progressive JavaScript frameworks for building user interfaces. Unlike other large frameworks, Vue is designed to be applied layer by layer from the bottom up. Vue's core library focuses only on the view layer, making it easy to get started and integrate with third-party libraries or existing projects. On the other hand, when combined with modern toolchains and various supporting libraries, Vue is perfectly capable of providing drivers for complex single-page applications (SPA).

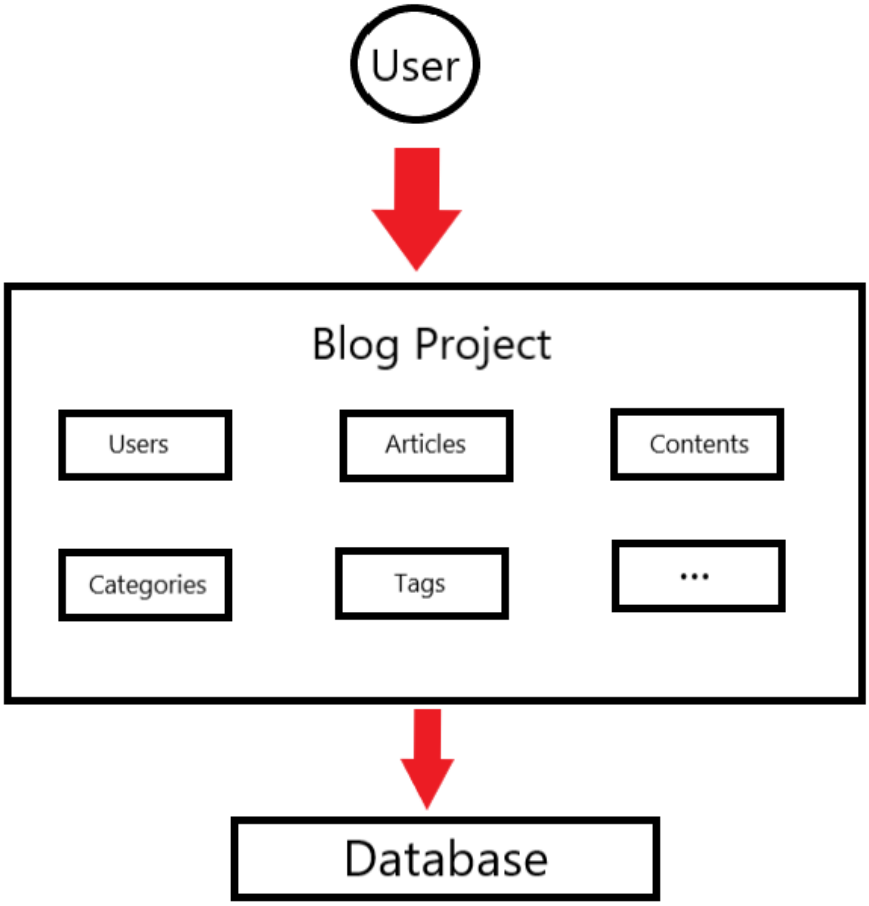
The backend we use is mainly composed of SpringBoot + Mybatisplus + Redis +MySQL framework.

SpringBoot: SpringBoot integrates common frameworks such as Mybatis+ SpringMVC, eliminating complex configuration.

Mybatisplus: Mybatisplus is a MyBatis enhancement tool, designed to enhance MyBatis on the basis of only do not change, to simplify development, improve efficiency

Redis: Redis supports data persistence, saving data in memory to disk, which can be reloaded for use upon restart. Redis not only supports simple key-value type data, but also provides the storage of list, set, zset, hash and other data structures.

MySQL: Back-end storage of various data needs to design and use the database; MySQL is an open-source relational database management system.

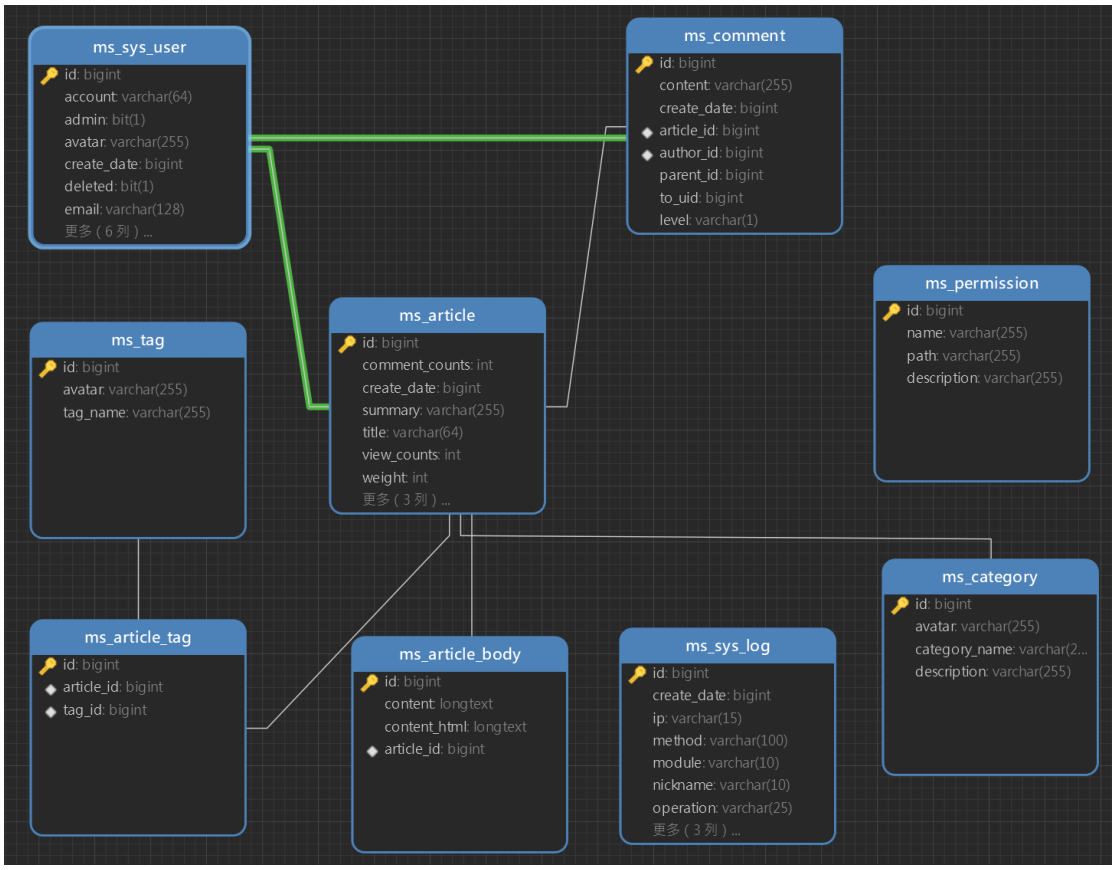


High-level Database Design

Tables: Article, Article_body, Article_tag, Tag, Categoty, Comment, Permission, Sys_log, Sys_User.

Details of the database:
Article contains article information: Id, Create_Time, Comment count, view cont, title, etc
Article_body: Contain the main body of the article.
Article_tag: Connect Table Article and Table Tag.
Tag: Article tags.
Category: Article categories.
Comment: Contain all comment information.
Permission: User's rights on the app.
Sys_log: Record users' operations on the app.(Just create the table. It will not be used util the app is certainly open to the users.)

Sys_user: Contain all the registered users' basic information.



Work Process

Product Backlogs

For the functions we would like to achieve in this product, we firstly defined two epics, 'Blog Management' and 'User Account Management'.

Blog management includes all interactions centered on the core element we have in this product —— blogs. We developed 4 features in this epic:

1. 'Main Page Components' which includes all functions in the main page.
2. 'Blog Full Text Page Components' which includes all functions in the blog full text page (this page shows up after clicking on one blog in the main page).
3. 'Blog posting' which includes functions to realize posting users' own blogs.
4. 'Blog Search' (not implemented and undecided) which allows users to search for blogs posted in this website.

User account management include all interactions centered on the user's account. We developed 3 features in this epic:

1. 'User Login' which enables users to log in to the website to gain advanced options such as posting and commenting using a registered account.
2. 'User Registration' which enables users to create an account in this website to login.
3. 'User Profile' (not implemented and undecided) which allows users to modify some related information of their accounts.

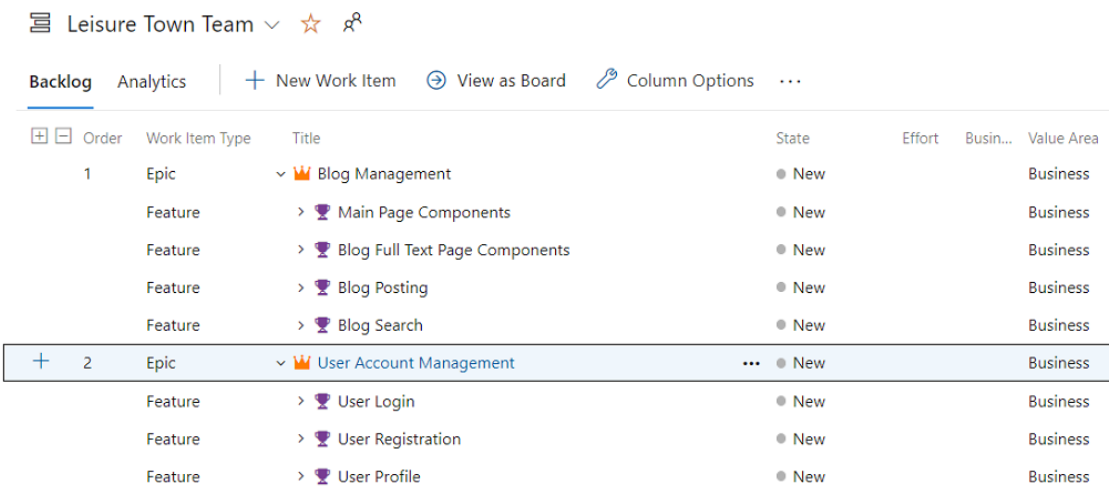


Fig. Product backlog overview

We groomed and accomplished 11 PBIs in total over the past 3 sprints.

Sprint 1 —— 2 PBIs

#1 Login

PRODUCT BACKLOG ITEM 19*

19 Login

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 1

Description

The function enables visitors to log in to Leisure Town by a registered account.

As a user, **I want to** log in to Leisure Town **so that** I can interact with the app functions.

Q: What should be done if the user doesn't have an account on this website?
A: A registration icon should be put in the login page to manage this.

Q: What should be done if the user types wrongly on his/her ID or password?
A: A reminder message should be displayed to inform users to input again.

Acceptance Criteria

Scenario 1: user with a registered account.
GIVEN the account is registered in the database **WHEN** the user finishes filling in the ID and password THEN the system will redirect to the main page of Leisure Town.

Scenario 2: user typing the ID or password wrongly.
GIVEN the info is wrong in terms of ID or password **WHEN** filling in the account info **THEN** an error message should be displayed to tell the user to try again because of wrong ID or password.

Scenario 3: user with missing information of the ID or password .
GIVEN the info (ID or password) is missing **WHEN** filling in the account info **THEN** an error message should be displayed to tell the user to try again because of missing content.

Details

Priority
1

Effort
5

Business Value

Value area
Business

Fig. Groomed documentation for PBI: Login

We divided this PBI into 7 tasks:

1.1 UI – login page (assigned to xinyi.huang). This task is to create the UI layout for the login interface. Login box requires the username and password, plus a submit button. A background image is also provided for aesthetics.

1.2 Front-end api (assigned to junhe.yan) - This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

1.3 Front-end script (assigned to xinyi.huang) - This task is to develop the front-end code to realize the login function.

1.4 Router – to main page (assigned to chuanzhi.su). This task is to develop the router to redirect to the main page after the login process succeeds.

1.5 Login interceptor (assigned to zhaoyu.xu) - The login interceptor is an element preventing operations that needs authenticated users' manipulation, such as posting and commenting, from getting influenced by an unauthenticated user. In other words, the guests are banned from these operations until they log in as a user.

1.6 Back-end api coding (assigned to ruowei.xing) - This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

1.7 Testing. (assigned to sirui.li) - This task is to test the usability and completion of this function.

#2 Registration

PRODUCT BACKLOG ITEM 35

35 Registration

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 1

Description

The function enables visitors to create an account in Leisure Town.

As a user, I **want to** have an account registered in Leisure Town **so that** I can have access to it and log in later.

The data type of ID and password should be a **string**.
The length of **ID** is **less than 10 characters**, **16** for **password**.

Q: Is there limitation on the ID and password input?
A: It is acknowledged as long as they are strings and within the defined maximum length.

Acceptance Criteria

Scenario 1: user with a legal input.
GIVEN the input is in a proper format **WHEN** the user finishes filling in **THEN** the system will redirect to the login interface and the database will create one new record.

Scenario 2: user with an illegal input.
GIVEN the input is in an unwanted format **WHEN** the user finishes filling in **THEN** an error message should be displayed to tell the user to try again because of illegal input format, too long ID or password or missing content.

Scenario 3: user with duplicate ID.
GIVEN the input ID conflicts with others' ID **WHEN** the user finishes filling in **THEN** an error message should be displayed to tell the user to try again and change the ID because of duplicate ID.

Details

Priority
1

Effort
5

Business Value

Value area
Business

Fig. Groomed documentation for PBI: Registration

We divided this PBI into 7 tasks:

2.1 UI – registration page (assigned to xinyi.huang). This task is to create the UI layout for the registration interface. Registration box requires the username, the nickname and password, plus a submit button. A background image is also provided for aesthetics.

2.2 Front-end api. (assigned to junhe.yan) - This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

2.3 Front-end script. (assigned to xinyi.huang) - This task is to develop the front-end code to realize the registration function.

2.4 Router – to main page (assigned to chuanzhi.su). This task is to develop the router to redirect to the main page after the registration process succeeds.

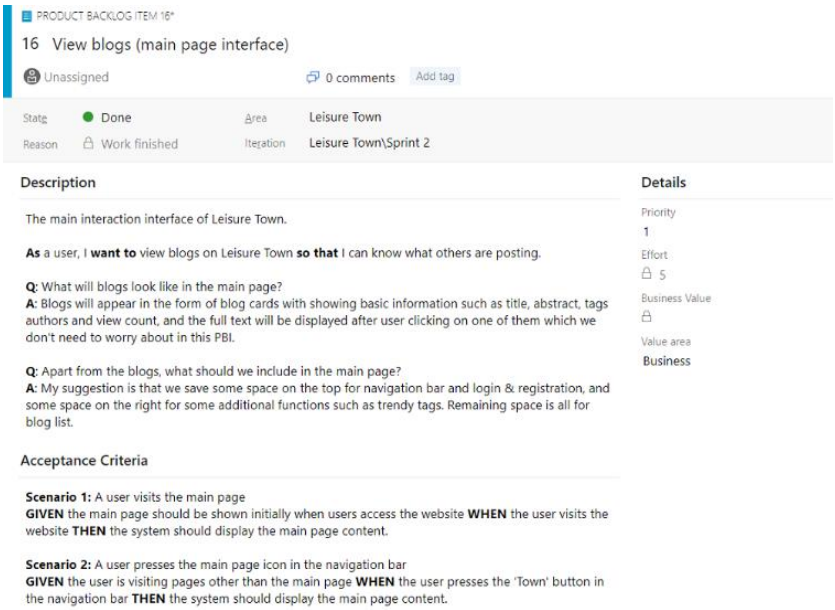
2.5 Back-end api coding. (assigned to ruowei.xing) -This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

2.6 Database initialization. (assigned to ruowei.xing) This task is to create corresponding tables in the schema to store information of user accounts.

2.7 Testing. (assigned to ruidong.shen) -This task is to test the usability and completion of this function.

Sprint 2 —— 4 PBIs

#3 View blogs (main page interface)



PRODUCT BACKLOG ITEM 16*

16 View blogs (main page interface)

Unassigned | 0 comments | Add tag

State: Done | Area: Leisure Town

Reason: Work finished | Iteration: Leisure Town\Sprint 2

Description

The main interaction interface of Leisure Town.

As a user, I **want to** view blogs on Leisure Town **so that** I can know what others are posting.

Q: What will blogs look like in the main page?
A: Blogs will appear in the form of blog cards with showing basic information such as title, abstract, tags authors and view count, and the full text will be displayed after user clicking on one of them which we don't need to worry about in this PBI.

Q: Apart from the blogs, what should we include in the main page?
A: My suggestion is that we save some space on the top for navigation bar and login & registration, and some space on the right for some additional functions such as trendy tags. Remaining space is all for blog list.

Acceptance Criteria

Scenario 1: A user visits the main page
GIVEN the main page should be shown initially when users access the website **WHEN** the user visits the website **THEN** the system should display the main page content.

Scenario 2: A user presses the main page icon in the navigation bar
GIVEN the user is visiting pages other than the main page **WHEN** the user presses the 'Town' button in the navigation bar **THEN** the system should display the main page content.

Details

Priority: 1
Effort: 5
Business Value:
Value area: Business

Fig. Groomed documentation for PBI: View blogs

We divided this PBI into 7 tasks:

3.1 UI – main page + blog component (assigned to xinyi.huang). This task is to create the UI layout for the main page. The main page consists of a navigation bar storing multiple options such as 'Town (Main page)', login and registration, blogs which take up most space, and some advanced functions such as trendy tags. The blog component consists of a title, an abstract, the author's name, the tags, view count, post time, etc.

3.2 Front-end api. (assigned to junhe.yan) This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

3.3 Front-end script. (assigned to xinyi.huang) This task is to develop the front-end code to realize the function to show posted blogs in the main page.

3.4 Router – to main page (assigned to chuanzhi.su). This task is to develop the router in the navigation bar, 'Town' icon, to redirect to the main page. It is useful when you want to go back to the main page when you are visiting other pages.

3.5 Back-end api coding. (assigned to lincheng.shi) This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

3.6 Database initialization. (assigned to lincheng.shi) This task is to create corresponding tables in the schema to store information of posted blogs.

3.7 Testing. (assigned to lincheng.shi) This task is to test the usability and completion of this function.

#4 Blog tags

PRODUCT BACKLOG ITEM 69*

69 Blog tags

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 2

Description

The element for marking blogs in particular fields.

As a user, I **want to** add tags on my blogs **so that** I can mark the topics of my blogs. I **also want to** see tags of other blogs in the main page **so that** I can get their topics briefly.

Q: How to define the tag?
A: The tag is a tool to mark the topic of a blog. For example, a tag named 'Travelling' is likely to be added in a blog talking about journeys. The tags are **pre-defined** to cover a broad range of possible blog topics.

Q: Where to place it in the blog component?
A: At the bottom of the blogs, under the abstract. It's in the form of a green brick, as our theme color is green.

Acceptance Criteria

Scenario 1: A login user adds a valid tag
GIVEN the user is authenticated **WHEN** the user ticks the tags he/she wants in his blog **THEN** the system should display the tags of the blog the user added in the blog component.

Scenario 2: A user sees other blogs' tags
GIVEN the user visits the website **WHEN** the user sees the blogs in the main page **THEN** the system should display the corresponding tags of each blog in the blog component.

Details

Priority
2

Effort
4

Business Value

Value area
Business

Fig. Groomed documentation for PBI: Blog tags

We divided this PBI into 6 tasks:

4.1 UI – tags in the blog component (assigned to xinyi.huang). This task is to create the UI layout for the tags in the blog component. It appears in green bricks to inform users of the information it contains.

4.2 Front-end api. (assigned to junhe.yan) This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

4.3 Front-end script. (assigned to zhaoyu.xu) This task is to develop the front-end code to realize the function to show tags in the blog component.

4.4 Back-end api coding. (assigned to ruidong.shen) This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

4.5 Database initialization. (assigned to ziyu.xiong) This task is to create corresponding tables in the schema to store information of blog tags.

4.6 Testing. (assigned to ziyu.xiong) This task is to test the usability and completion of this function.

#5 Blog authors

PRODUCT BACKLOG ITEM 70*

70 Blog authors

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 2

Description

The element to demonstrate the author of this blog.

As a user, I **want to** view the author of blogs **so that** I can get who write pretty blogs, or awful ones.

Q: Where to place it in the blog component?
A: At the bottom of the blogs, on the left of tags.

Q: What should the author name be, the username or the nickname?
A: The nickname.

Acceptance Criteria

Scenario 1: A login user posts blogs
GIVEN the user is authenticated **WHEN** the user posts a blog **THEN** the system should display the user's nickname in the specific position of the blog component.

Scenario 2: A user sees others' blogs in the main page
GIVEN the user visits the website **WHEN** the user sees the blogs displayed in the main page **THEN** the system should show the author name for each displayed blog to the user.

Details

Priority
2
Effort
3
Business Value
Value area
Business

Fig. Groomed documentation for PBI: Blog authors

We divided this PBI into 5 tasks:

5.1 UI – authors in the blog component (assigned to xinyi.huang). This task is to create the UI layout for the authors in the blog component. It appears beside the tags to show the author of the blog.

5.2 Front-end api (assigned to junhe.yan). This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

5.3 Front-end script (assigned to zhaoyu.xu). This task is to develop the front-end code to realize the function to show author names in the blog component.

5.4 Back-end api coding (assigned to ruidong.shen). This task is to refine the back-end code in the blog function. As the author is already stored when posting, some extra methods to extract and show the author of the blog are implemented and added.

5.5 Testing (assigned to ziyu.xiong). This task is to test the usability and completion of this function.

#6 Log out

PRODUCT BACKLOG ITEM 68*

68 Log out

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 2

Description

The function to allow user to take back his account from the website for security.

As a user, I **want to** cancel login **so that** I can log off when I don't want to surf the web anymore.

Q: Where should we put the log out icon?
A: It should be put under the login icon (it becomes an avatar after login) as it belongs to the login feature. Normally it is hidden, but when clicking on the avatar icon on the right-top, the log-out option is shown in a second level.

Acceptance Criteria

Scenario 1: A login user
GIVEN the user has logged in **WHEN** the user presses the cancel login button **THEN** the system should log out the user account and change back to the original login icon.

Scenario 2: Not logged-in user
GIVEN the user hasn't logged in **THEN** the system should not provide the log-out option in specific location, i.e., there should only be a login icon to allow users to log in.

Details

Priority
2
Effort
2
Business Value
Value area
Business

Fig. Groomed documentation for PBI: Log out

We divided this PBI into 5 tasks:

6.1 UI – log out (assigned to xinyi.huang). This task is to create the UI layout for the log out option embedded under the user's avatar on the right-top of the interface.

6.2 Front-end api (assigned to junhe.yan). This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

6.3 Front-end script (assigned to zhaoyu.xu). This task is to develop the front-end code to realize the function to cancel users' login status and change back to the not logged-in operation style.

6.4 Back-end api coding (assigned to ziyu.xiong). This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

6.5 Testing (assigned to ziyu.xiong). This task is to test the usability and completion of this function.

Sprint 3 —— 5 PBIs

#7 Blog view count

PRODUCT BACKLOG ITEM 98

98 Blog view count

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 3

Description

The element in blog component to show the number of views for one blog.

As a user, I **want to** see the view count of a blog **so that** I can feel how many users view one blog to get how popular each blog is.

Q: In what condition should we add the view count?
A: If the user **clicks on one blog** to see the details, we increment the view count.

Acceptance Criteria

Scenario 1: A user sees one blog's view count
GIVEN the user visits the website **WHEN** the user sees a blog in the main page **THEN** the system should display the view count in the blog component.

Scenario 2: A user clicks on one blog to see full text
GIVEN the user is visiting the main page **WHEN** the user clicks on one blog to see the full text **THEN** the system should increment the view count by 1.

Details

Priority: 3
Effort: 2
Business Value:
Value area: Business

Fig. Groomed documentation for PBI: Blog view count

We divided this PBI into 6 tasks:

7.1 UI – view count in the blog component (assigned to xinyi.huang). This task is to create the UI layout for the view count in the blog component. It appears with a eye icon along with the number of views.

7.2 Front-end api (assigned to junhe.yan). This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

7.3 Front-end script (assigned to xinyi.huang). This task is to develop the front-end code to realize the function to show tags in the blog component.

7.4 Back-end api coding (assigned to ruowei.xing). This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

7.5 Database refinement – add view count column (assigned to ruowei.xing). This task is to refine the blog (article) table in the schema by adding the column of view count for further reference and use.

7.6 Testing (assigned to ziyu.xiong). This task is to test the usability and completion of this function.

#8 Trendy tags

PRODUCT BACKLOG ITEM 81*

81 Trendy tags

Unassigned 0 comments Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 3

Description

The function to arrange the most frequently used tags together in one section.

As a user, I **want to** see the trendy tags **so that** I can get what hot topics are in current context.

Q: How to define the trendy tags?
A: the tags that are used most among all tags, i.e., tags that most blogs use.

Q: How many tags we need to show in the trendy tag part?
A: 6 tags would be appropriate.

Acceptance Criteria

Scenario 1: A user wants to see trendy tags
GIVEN the user is visiting the website **WHEN** the user tries to see trendy tags **THEN** the system should display current trendy tags in the particular position in the main page.

Details

Priority
3

Effort
4

Business Value

Value area
Business

Fig. Groomed documentation for PBI: Trendy tags

We divided this PBI into 5 tasks:

8.1 UI – in the main page (assigned to xinyi.huang). This task is to create the UI layout for the trendy tag function in the main page. It lies on the right of the blogs.

8.2 Front-end api (assigned to junhe.yan). This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

8.3 Front-end script (assigned to xinyi.huang). This task is to develop the front-end code to realize the function to show the trendy tags in assigned position,

8.4 Back-end api coding (assigned to ruidong.shen). This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

8.5 Testing (assigned to ruidong.shen). This task is to test the usability and completion of this function.

#9 Blog full text

PRODUCT BACKLOG ITEM 93*

93 Blog full text

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 3

Description

An individual page to show the full text of the blog, appears after the user clicks on one blog in the main page.

As a user, I **want to** see the whole text of the blog **so that** I can see the blog in detail and make further interactions like comment.

Q: What is the full text interface look like?
A: The full text of the blog along with corresponding information should be included, following the comment list of the blog.

Acceptance Criteria

Scenario 1: A user wants to see the full text
GIVEN the user is visiting the website **WHEN** the user clicks on a posted blog **THEN** the system should display the whole text of the blog and the comment list.

Details

Priority
2

Effort
5

Business Value

Value area
Business

Fig. Groomed documentation for PBI: Blog full text

We divided this PBI into 6 tasks:

9.1 UI – blog detail page (assigned to xinyi.huang). This task is to create the UI layout for the blog detail page. It consists of a title, information of the author, the full text of the blog, 'the end of the blog' brick and ticked tags and category.

9.2 Front-end api (assigned to junhe.yan). This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

9.3 Front-end script (assigned to xinyi.huang). This task is to develop the front-end code to realize the function to show the full text and other element in this page.

9.4 Router – to full text page (assigned to chuanzhi.su). This task is to develop the router from the main page to the full text page of the clicked-on blog.

9.5 Back-end api coding (assigned to lincheng.shi). This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

9.6 Testing (assigned to lincheng.shi). This task is to test the usability and completion of this function.

#10 Comment

PRODUCT BACKLOG ITEM 40*

40 Comment

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 3

Description

The function to leave a message under a blog.

As a user, I **want to** comment on someone's blog **so that** I can express my opinion to the blogger.

Q: What should the comment component include?
A: The comment No., the nickname, the avatar and the comment content of the commenter.

Q: Should we implement a multi-level comment?
A: Yes, we should realize the function of **commenting on another comment** as a second-level comment to manage the communication not only to the authors, but also to the commenters.

Q: What does the second-level comment look like?
A: It lies under the main comment, uses a smaller size of all included elements, and has an "@the commenter you reply to" text in the beginning of the content to show who you are commenting on.

Acceptance Criteria

Scenario 1: A login user sends a comment
GIVEN the user is authenticated and the comment content is not empty **WHEN** the user clicks on the 'comment' button **THEN** the system should send the comment to the list of comments for this blog.

Scenario 2: A not logged in user sends a comment
GIVEN the user hasn't logged in **WHEN** the user types in a comment and clicks on the 'comment' button **THEN** a error message indicating comment failure should be displayed to remind user.

Details

Priority
3

Effort
5

Business Value

Value area
Business

Fig. Groomed documentation for PBI: Comment

We divided this PBI into 6 tasks:

10.1 UI – comment (assigned to xinyi.huang) This task is to create the UI layout for the comments. It consists of the total number of comments, and for each comment it has the commenter's nickname and avatar, the comment number and the comment content. A second-level comment UI is also implemented, with same structure but in a smaller size.

10.2 Front-end api (assigned to junhe.yan). This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

10.3 Front-end script (assigned to xinyi.huang). This task is to develop the front-end code to realize the function to show the comments of the blogs in the blog full text page.

10.4 Back-end api coding (assigned to sirui.li). This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc.

10.5 Database initialization (assigned to sirui.li). This task is to create corresponding tables in the schema to store information of comments.

10.6 Testing (assigned to sirui.li). This task is to test the usability and completion of this function.

#11 Post blogs

PRODUCT BACKLOG ITEM 20*

20 Post blogs

Unassigned | 0 comments | Add tag

State	Done	Area	Leisure Town
Reason	Work finished	Iteration	Leisure Town\Sprint 3

Description

The function to allow user to post their own blogs on the website.
As a user, I **want to** post blogs **so that** I can express myself to all users in this website.

Q: Where should we put the post icon in the main page?
A: Also in the navigation bar, but away from other options as it's a core function. Let's say put it on the left of login/registration.

Q: What should we include in the post interface? Should we make it **multi-level**?
A: Yes, we can make it multi-level for posting to better organize the overall process.
In the initial interface, 4 things should be included: **1. Title entry bar**, **2. Content entry bar**, **3.** It's better if we can find **a text editor** containing such as **bold** and *Italian* options to enable users to polish their text, **4. Post and Cancel icons.**
After the user fills in the title and the content and clicks on the post icon, a window will jump out for advanced options and the following elements should be included: **1. the abstract** to be shown in the main page, **2. add tags and category** it belongs to, **3. Confirm Post and Cancel icons.**

Acceptance Criteria

Scenario 1: A login user wants to post a blog
GIVEN the user is authenticated and all needed elements are properly filled in **WHEN** the user tries to post a blog **THEN** the system should display the blog in the main page in the blog list with corresponding elements (title, abstract, tag, author, view count, post time etc.).

Scenario 2: A login user wants to post a blog with missing element(s).
GIVEN the user is authenticated but one or more element needed for posting is null (Title, content, abstract, tag or category) **WHEN** the user tries to post a blog **THEN** the system should display a message informing the user to complete the needed element respectively.

Scenario 3: A login user doesn't want to post a blog anymore
GIVEN the user is authenticated **WHEN** the user presses the cancel button in the post interface **THEN** the system should display a message for confirmation. After confirmation, the system shall redirect to the main page.
(if clicking on the cancel button in the advanced option page, it'll only jump back to the previous post interface with title and content.

Details

Priority
2
Effort
5
Business Value
Value area
Business

Fig. Groomed documentation for PBI: Post blogs

We divided this PBI into 7 tasks:

11.1 UI – post blog in the main page (assigned to xinyi.huang). This task is to create the UI layout for posting blog function in the main page. It is put in the navigation bar and has a pen beside the function name to demonstrate the functionality.

11.2 UI – interface for writing blogs (assigned to xinyi.huang). This task is to create the UI layout for interface for writing blogs. It is two-level with the first level being a page to write the title and full text, and the second level being a jump-out window to write the abstract and choose tags and category. Icons like post or cancel posting are also designed for flexibility.

11.3 Front-end api (assigned to junhe.yan). This task is to develop the front-end api to connect to the back-end with a port and code the required data for this function.

11.4 Front-end script (assigned to chuanzhi.su). This task is to develop the front-end code to realize the function to post blogs, choose tags and category and other advanced options in posting function.

11.5 Routers (assigned to chuanzhi.su). This task is to develop the routers encountered in posting. Firstly, the router from the main page to the posting page is implemented. Next, the router from the posting page to the main page when the blog editing process is cancelled is implemented. Finally, the router from the posting page to the main page after the posting process succeeds is implemented.

11.6 Back-end api coding (assigned to ruowei.xing). This task is to develop the back-end code for the function, including java files such as controller, mapper, service and etc. It also appends the posted blogs into the table storing all blogs in the website.

11.7 Testing (assigned to ziyu.xiong). This task is to test the usability and completion of this function.

During the preparation phase for this product, we also outlined other functions such as search and profile function that remain not groomed and implemented, and we'll discuss to decide whether to accomplish, refine or desert each of them in further sprints based on the realization difficulty, member's skillset and degree of relevance with our product.

Sprint Backlogs

- Sprint 1

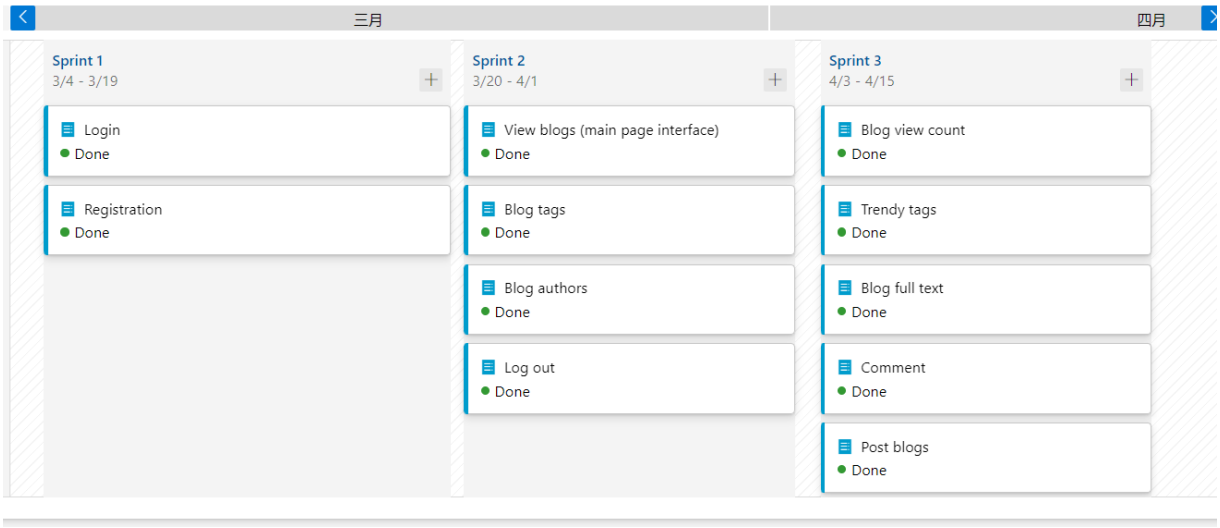
  In the first sprint, we decided to finish two PBIs of login and configuration and the front-end of the homepage. We firstly developed in a wrong way, we decided to assign several people to do the particular PBIs. However, most of us do not so familiar with springboot, so we get several same work in the first week. In the second week, we separated into front-end and back-end to complete the project. We finish the front-end and back-end and database of each PBI separately, but still have some problems of interface between login and homepage which should be improved in the next sprint in the end of sprint1.

- Sprint 2

In the second sprint, we planned to finish four PBIs, including: viewing the blogs' detail in a new page jumped by clicking the title in homepage, log out function, and the tags and authors of the blogs which will be used in many future PBIs. In this sprint we fix the bug in the sprint1 and complete all the PBIs in this sprint and some functions and also be used in sprint3. In addition we also submit all of our code on the GitHub for easy communication.

- Sprint 3

  In the third sprint, we planned to finish five PBIs, including: add comment to the blog and other's comment, blog full text, post blogs with different tags, trendy tags, view count of the blogs. Cause the most of the front-end and some functions are finished in the previous sprints, so we complete the PBIs in advance and focus on the report.



Definition of Done

A PBI can be marked done if it satisfies all following conditions:

1. The scripts for both the front-end and back-end are well written and can run without any error. Necessary data is stored in the database properly.
2. All acceptance criteria in the PBI document are met. Comprehensive test cases are conducted and passed to ensure to usability of the function in the product. (Back-end uses Postman and Front-end tests on the localhost)
3. In each stage, well-rounded corner cases are considered and managed to avoid unexpected situation happening when using. Besides, possible

relation and conflicts with existing PBIs are also considered comprehensively.

4. The connection between the front-end and back-end is established and the transmission of data between two end is successful. (Database insertion, parameter pass, etc.)
5. The accomplished code assigned to each group member is uploaded and integrated into the Git.

A task can be marked done if it satisfies all following conditions:

1. The scripts for either the front-end or back-end is well written and can run without any error. Necessary data is stored in the database properly.
2. Possible interactions with other tasks in the same PBI are considered and realized. Active communication with others in charge of related tasks is managed to avoid error in the integration phase.
3. Active discussion with the product owner is demonstrated to get a comprehensive view of the task job. The result of the task should be consistent with the thought of the product owner and group members that is proposed in the grooming session.
4. The assigned work is uploaded into the Git for others to check.

Sprint1:

Sprint Goal	Sprint Duration
1. build the springboot framework 2. complete PBIs of login and registration and front-end of homepage	Start Date: 4th May, 2022 End Date: 19th May, 2022

Group Members	1	2	3	4	5	6	7	8	9
Planned Effort(%)	11	11	11	11	11	11	11	11	11
Completion(%)	100	100	100	100	100	100	100	100	100

*The order is the same as the member list above

Sprint2:

Sprint Goal	Sprint Duration

1. improve the functions and UI design in Sprint2 2. complete 4 PBIs of blog list, tags, authors and logout	Start Date: 20th May, 2022 End Date: 1th April, 2022

Group Members	1	2	3	4	5	6	7	8	9
Planned Effort(%)	11	11	11	11	11	11	11	11	11
Completion(%)	100	100	100	100	100	100	100	100	100

Sprint3:

Sprint Goal	Sprint Goal
1. improve the functions and UI design in previous Sprints 2. complete 5 PBIs of view count, trendy tags, blog detail, comment and posting blogs 3. focus on report writing	Start Date: 3th April, 2022 End Date: 15th April, 2022

Group Members	1	2	3	4	5	6	7	8	9
Planned Effort(%)	11	11	11	11	11	11	11	11	11
Completion(%)	100	100	100	100	100	100	100	100	100

Comments:

- There are 11 PBIs are completed in the first three sprints, the functions can all run successfully, but there are still some aspects need to be i proved in the following sprints.
- The data transform successfully between froun-end and back-end, and data can be added or deleted successfully in the database.
- All the members in the group are work hard and finish the task they were assigned, and study for the project positively.

GitHub Upload

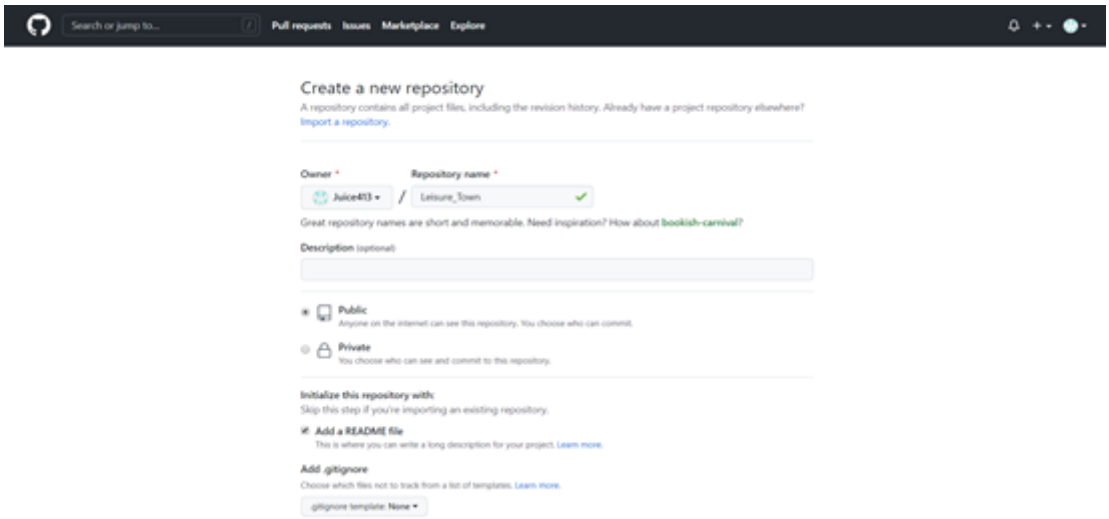
Repository link

[Link of the GitHub]

Uploading steps

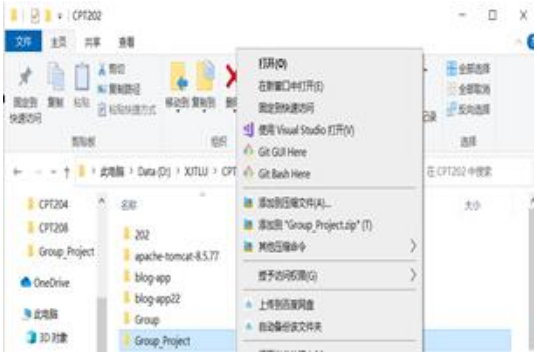
Upload Code to GitHub through Git on Windows

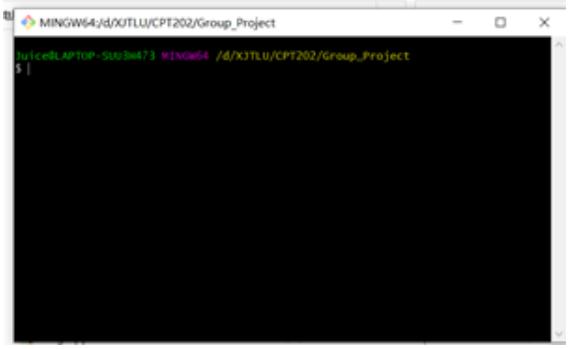
Creating a New Repository on GitHub



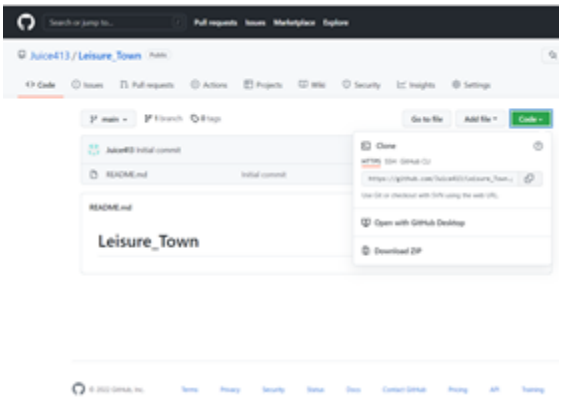
The repository will store the code we are about to upload, set the status to public, and allow team members to participate in code editing. Check Add a README File, and a README file will be automatically created in our project directory to explain our project.

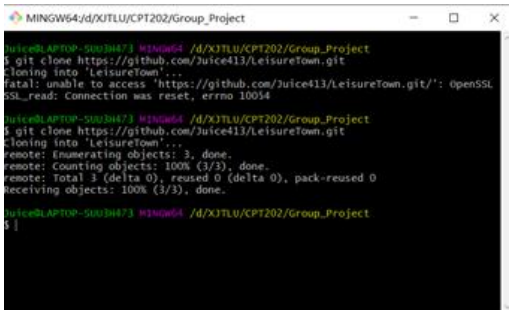
Open the project folder you want to upload, right-click the folder and select Git Base Here



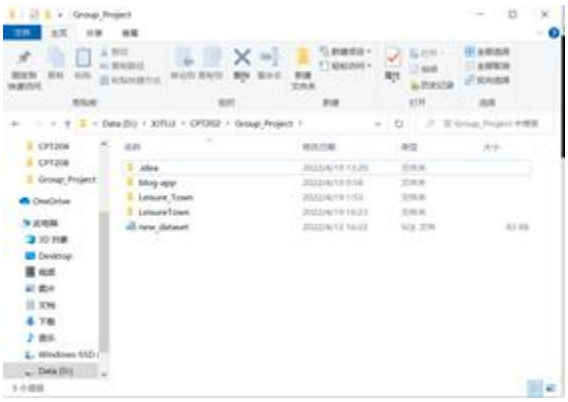


Go to the GitHub repository and select the repository address to copy it, enter git clone "repository address" in GIT command window





A folder with the same name as your repository will appear in your project folder



Next, you need to copy the files you want to upload to this folder and type in the GIT command window:
cd folder_name
go to the file location

```
Juice@LAPTOP-SUU3H473 MINGW64 /d/XJTLU/CPT202/Group_Project
$ cd LeisureTown
```

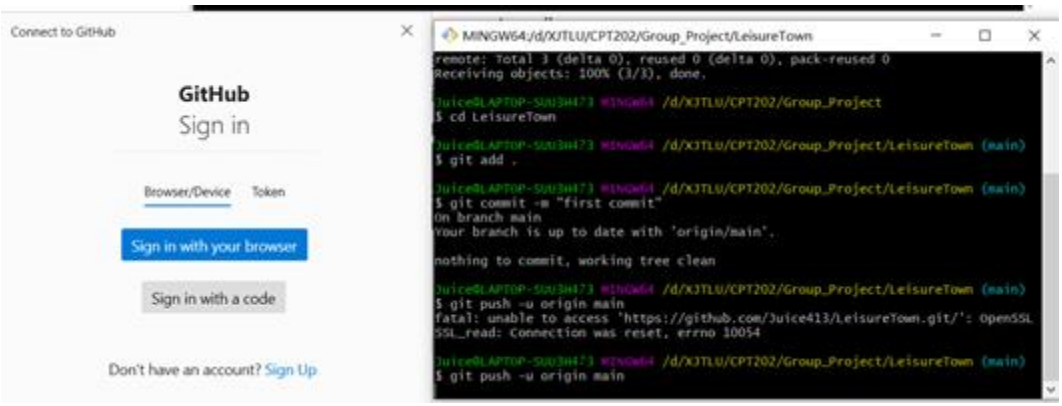
Type : git add. , enter (note that there is a '. ', this dot cannot be omitted)

```
Juice@LAPTOP-SUU3H473 MINGW64 /d/XJTLU/CPT202/Group_Project/LeisureTown (main)
$ git add .
```

Type : git commit -m "commit instructions"
(The purpose of submitting the explanation is to make myself or others clear about the purpose of uploading the file this time. Since the project may be modified later, the explanation of each modification is added for the convenience of reviewing the operation)

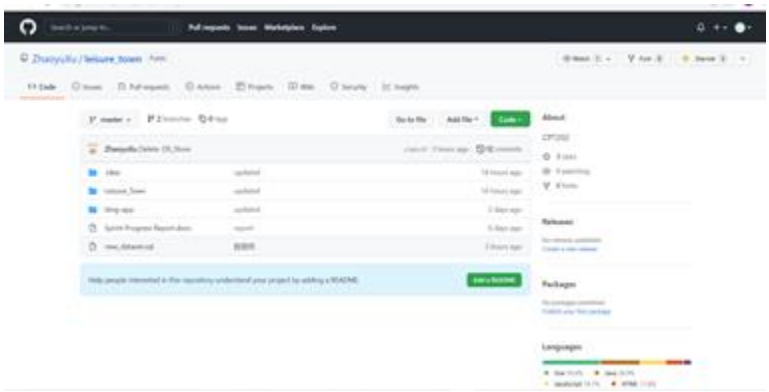
```
Juice@LAPTOP-SUU3H473 MINGW64 /d/XJTLU/CPT202/Group_Project/LeisureTown (main)
$ git commit -m "first commit"
On branch main
```

Type : git push -u origin main



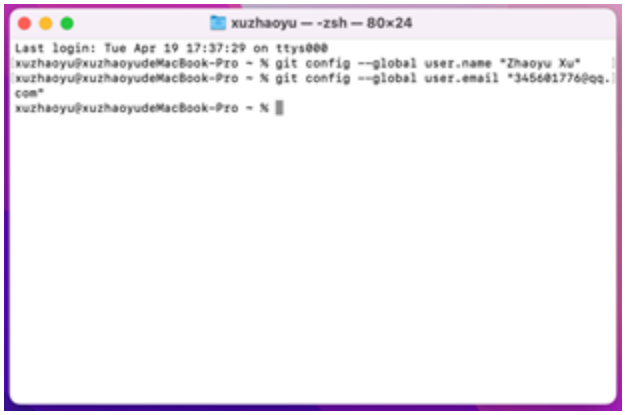
When executing this command for the first time, you will be asked to confirm your Github account password.

When the file is uploaded, go to your GitHub repository to view the file you uploaded.

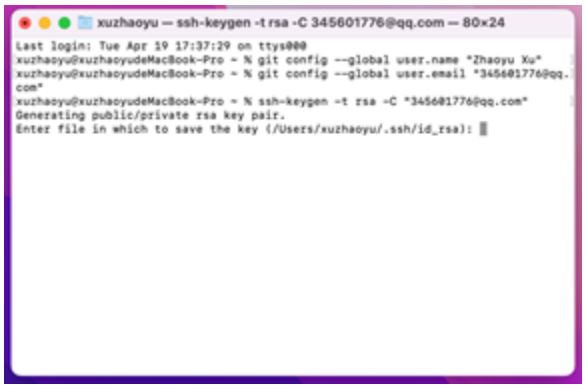


Upload Code to GitHub through Git on macOS

After installing the Git, open the Terminal and type: “git config --global user.name “your_name””and “git config --global user.email “your_email”” to configure your information

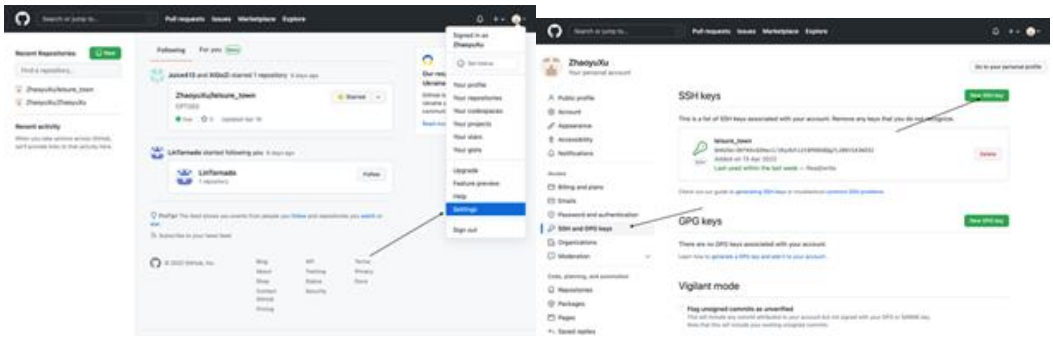


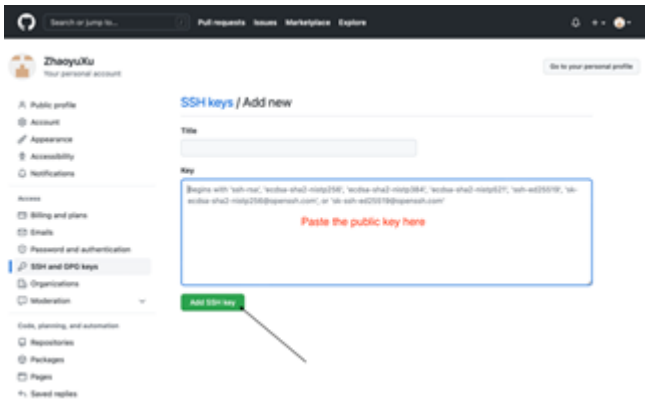
Type : “ssh-keygen -t rsa -C "your_email"” to generate a public key, then press **Return**



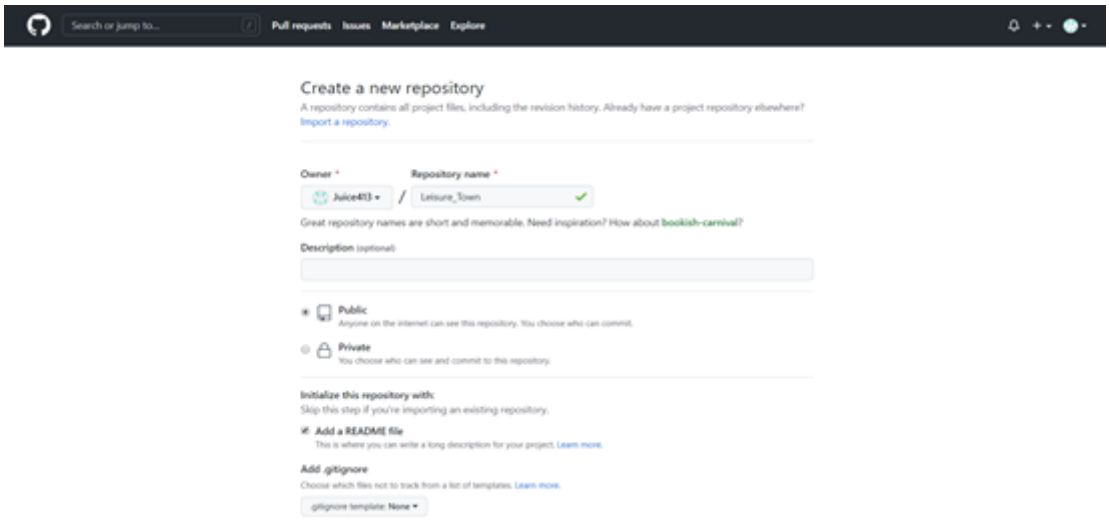
Press **Return** again and set your password for the key

Copy the public key and go to the GitHub to paste your public key





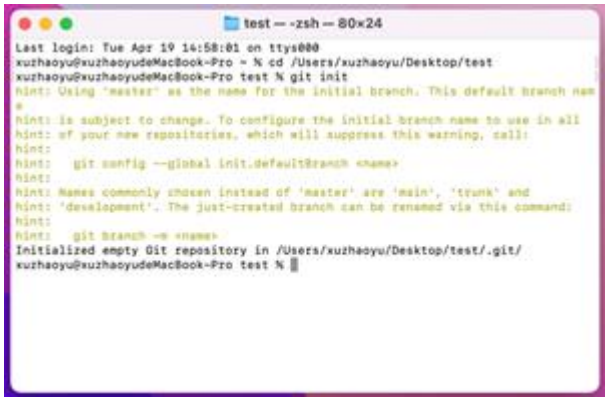
Creating a New Repository on GitHub



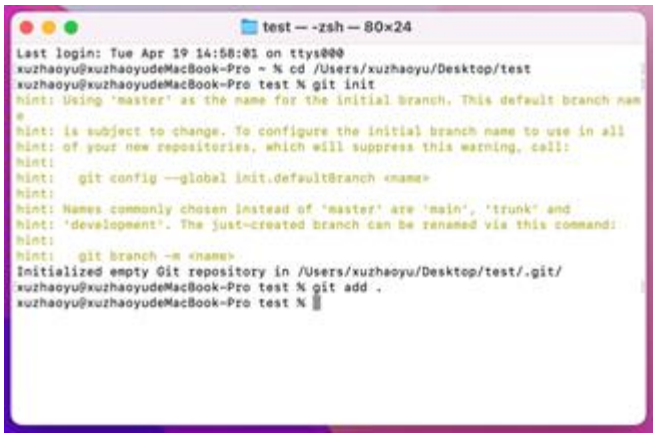
The repository will store the code we are about to upload, set the status to public, and allow team members to participate in code editing. Check Add a README File, and a README file will be automatically created in our project directory to explain our project.

Open the Terminal, type “cd” + “ ” and move the project folder into Terminal window, then press Return. It will into the folder.

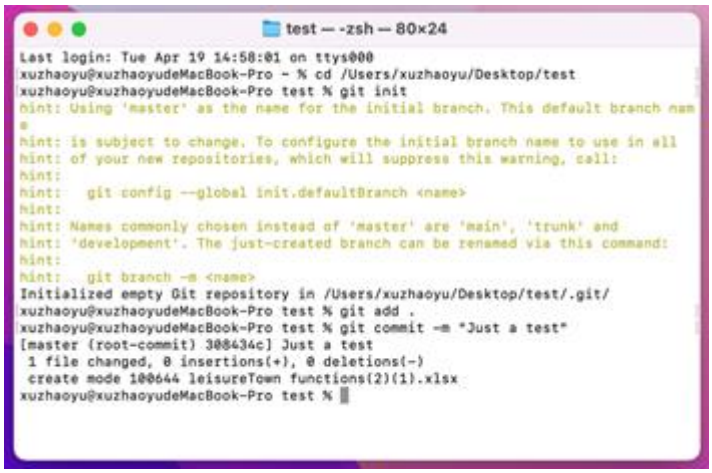
Type: “git init”, then press Return



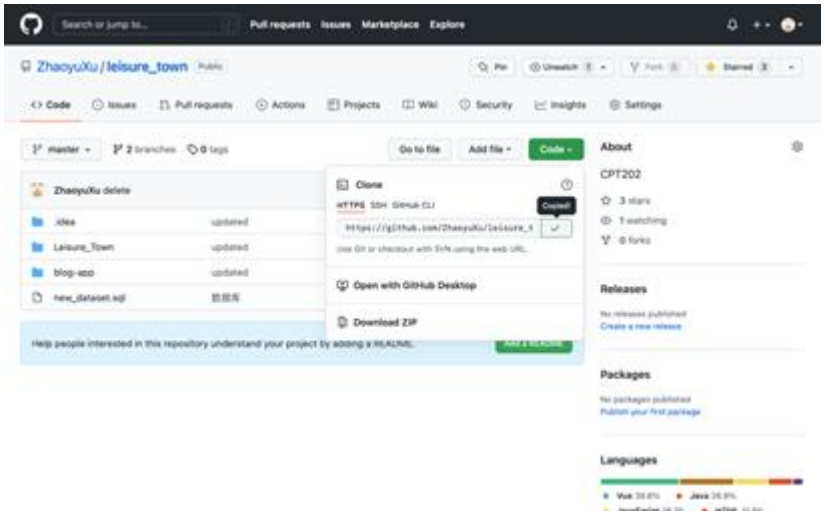
Type: “git add .” to add whole folder, then press Return



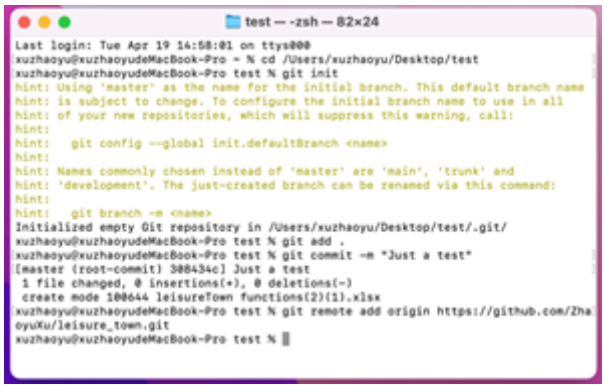
Type: “git commit -m “xxx””, to add an update record, then press Return



Go to the GitHub repository, copy the address



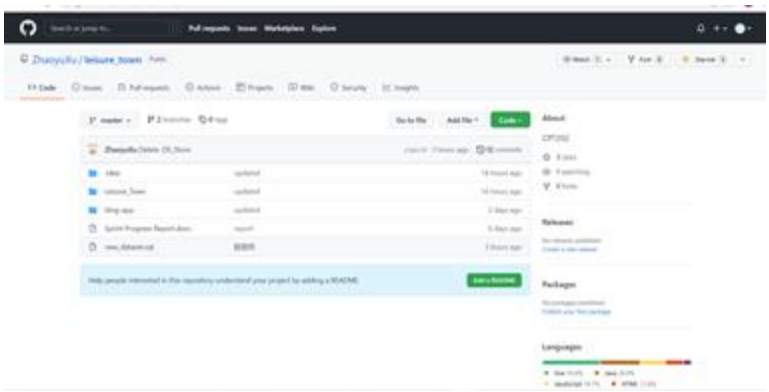
Back to the Terminal, type: “git remote add origin [Copied address]”, then press Return



Type: "git push origin master", then press Return

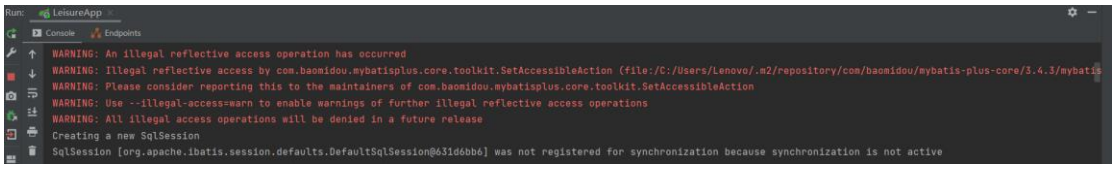
Input your password for the key, then press return

When the file is uploaded, go to your GitHub repository to view the file you uploaded.



Development Environment

We fully support our web app on Windows7 or later, MacOS and Linux. It can be displayed on Google Chrome, Miscrosoft Edge, Mozilla Firefox and so on. You will need to run our web app on a java compiler. We recommend using IntelliJ IDEA (hereinafter referred to as IDEA), which we used during development. Besides, the jdk is recommended to use version 13 or lower. Otherwise, the IDEA console may report an error as shown in the figure below when refershing the page, but this will not affect the normal running of our program.



Durning our development, we installed following software or libraries:

- Front-end:
  - Vue
  - element.ui
  - Node.js
  - mavon-editor
- Back-end:
  - SpringBoot
  - Maven
  - MyBatisPlus
  - Redis
- Database:
  - MySQL
- IDE:
  - IntelliJ IDEA
- Tools:
  - HBuilder X
  - Postman

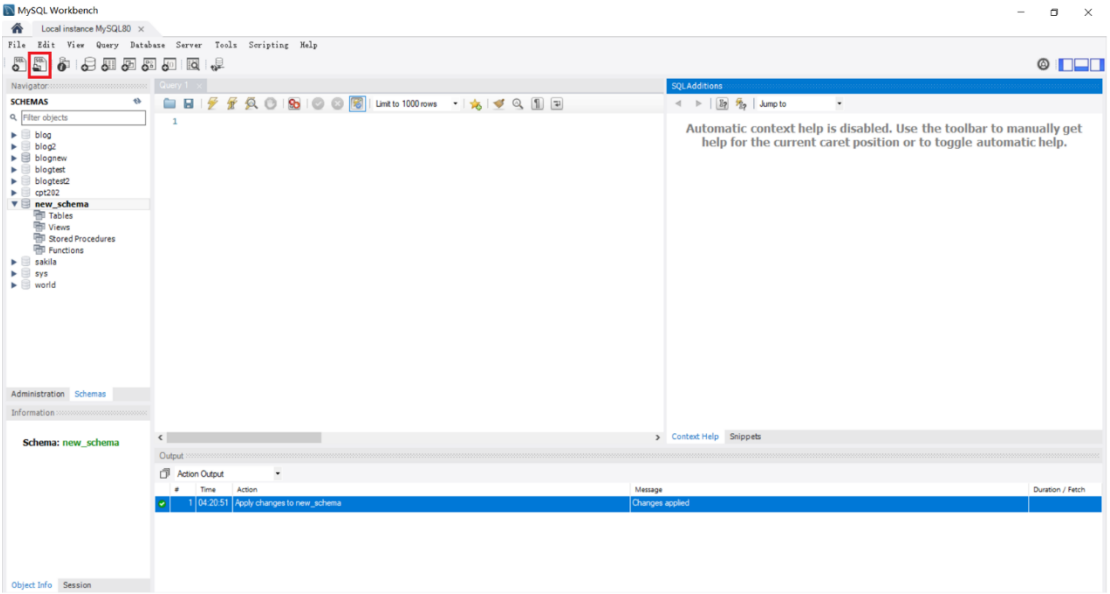
Production Environment

The first release of our product uses SpringBoot to run our local server, MyBatisPlus to connect to local database which is MySQL and the web pages are loaded by Vue framework.

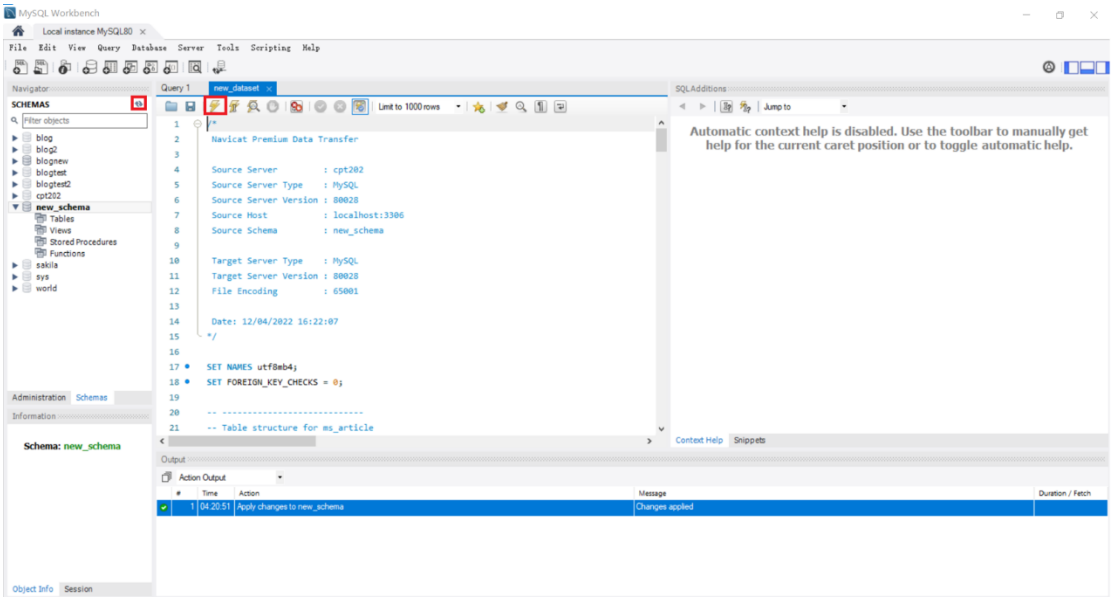
In addition to being on learningmall, you can also access our code and redis from GitHub (https://github.com/ZhaoyuXu/leisure_town). You can clearly see that our back-end folder is "Leisure_Town", the front-end folder is "blog-app", the database file is "new_dataset.sql" and the required redis is in the "redis1" folder (Windows version) or "redis2" folder (MacOS version).

Please follow the instructions below to set up the environment:

- Download two main projects (front-end: folder "blog-app", back-end: folder "Leisure_Town"), database file ("new_dataset.sql") and redis (folder "redis1" or folder "redis2") from GitHub link shown above.
- Install Node.js from https://nodejs.org/en/download/
- Start MySQL Workbench
    - Create a new schema called whatever you like (we recommend calling "new_schema") and double-click to select the schema.
    - Click "Open a SQL script file in a new query tab" and open our database file "new_dataset.sql".



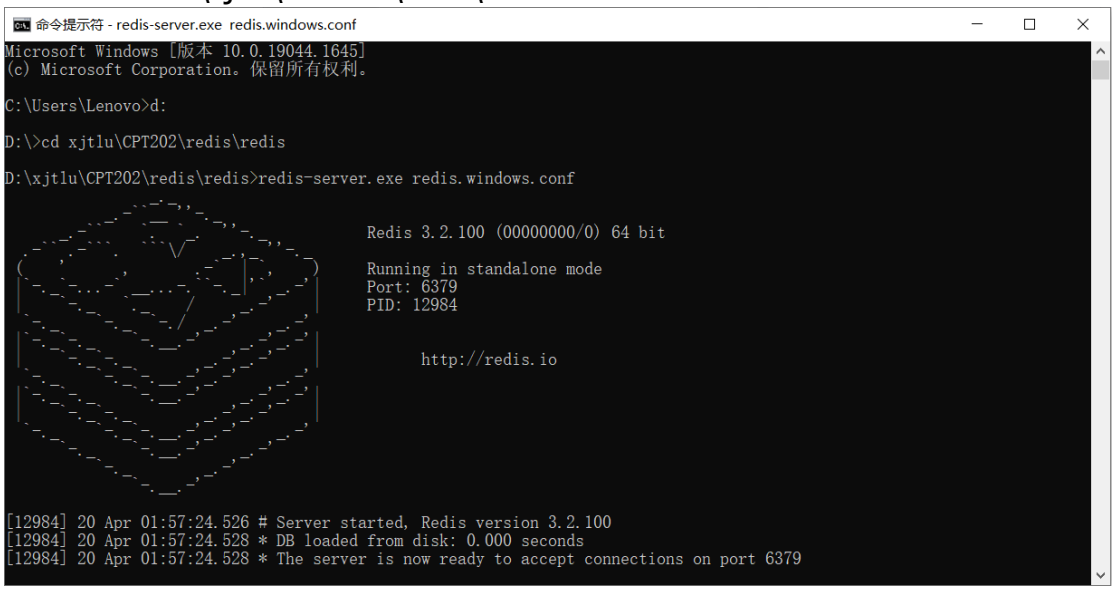
    - Click "Execute the selected portion of the script or everything, if there is no selection" and then refresh schema.



- Start "redis-server.exe" in folder "redis1" (Windows version) or folder "redis2\src" (MacOS version) which is downloaded in our Github link. (Don't forget this step, otherwise many functions will be unavailable if you can't register and log in!!!)

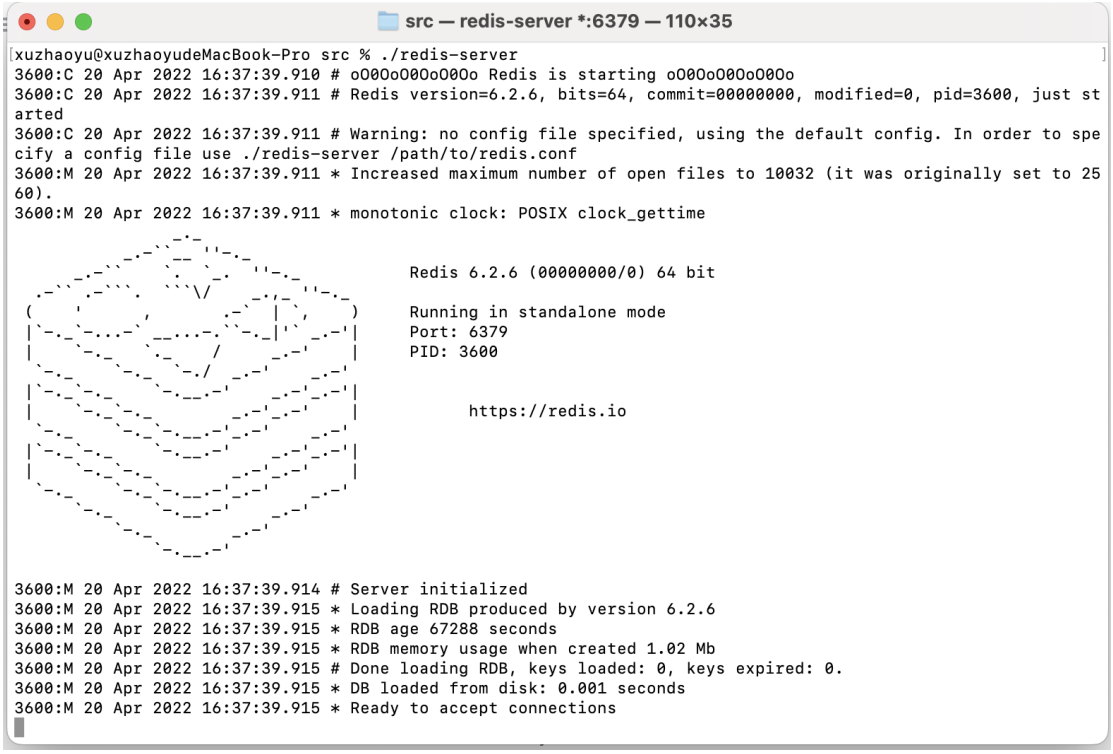
If you can't open redis successfully by clicking "redis-server.exe" directly:
For Windows:

- Open "cmd".
- Use the command to enter the folder "redis1" you downloaded from GitHub.
- Enter the command "redis-server.exe redis.windows.conf" and execute it.
- Here is an example: my "redis-server.exe" is in D:\xjtlu\CPT202\redis\redis



For MacOS:

- o Open "Terminal".
- o Use the conmmand to enter the folder "redis2\src" you downloaded from GitHub.
- o Enter the command "./redis-server" and execute it.
- o Here is an example: my "redis-server.exe" is in /usr/local/redis-6.2.6/src

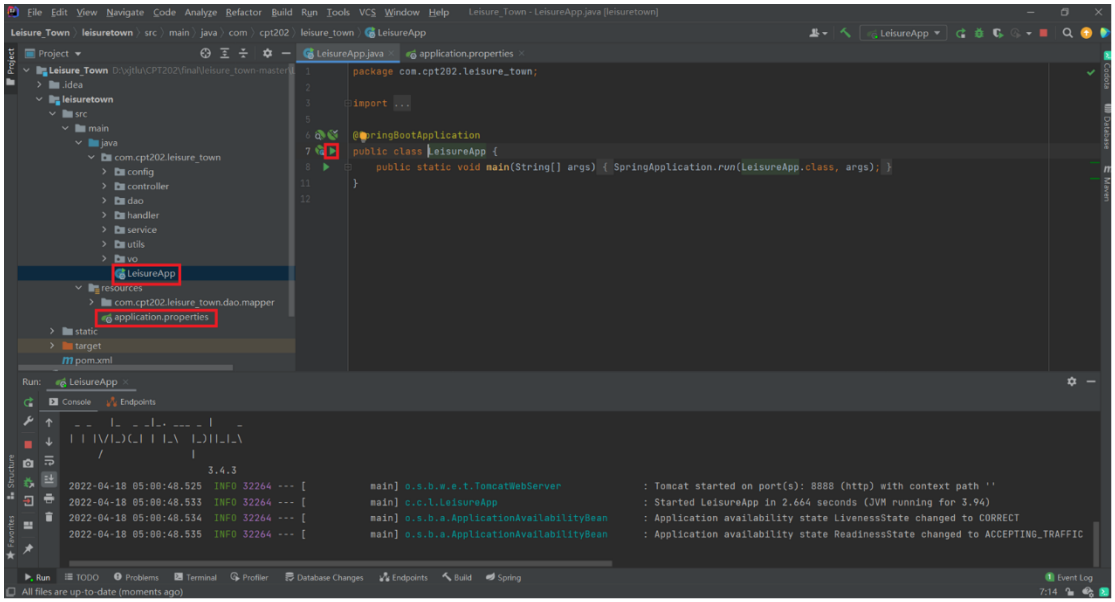


If you still have some problems about it in MacOS, please follow this blog:
https://blog.csdn.net/weixin_45037570/article/details/118500615

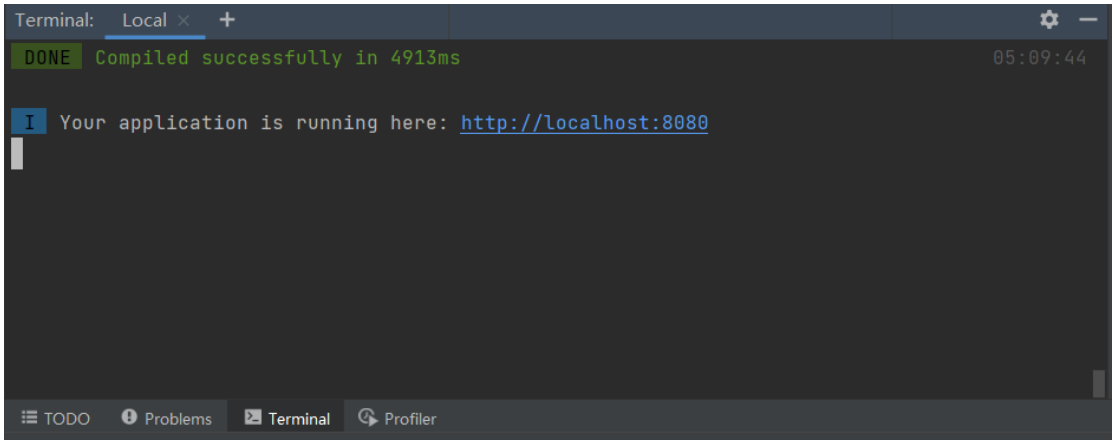
For Linux:

The steps are similar to those of MacOS. You can also check out with this blog: https://blog.csdn.net/qq_38728790/article/details/82703308?spm=1001.2101.3001.6661.1&utm_medium=distribute.pc_relevant_t0.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-1.pc_relevant_default&depth_1-utm_source=distribute.pc_relevant_t0.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-1.pc_relevant_default&utm_relevant_index=1

- Run back-end project "Leisure_Town"
  - o Use IntelliJ IDEA to open the project.
  - o Install related libraries and plugins "MyBatisX" and "Lombok".
  - o Go to "application.properties" file and change the "username" and "password" to your own MySQL account and change the "url" to your own schema which is created earlier. For example, my localhost is "3306" and my schema is called "new_schema", so url is "jdbc:mysql://localhost:3306/new_schema".
  - o Run the SpringBoot project by "LeisureApp.java".



- Run front-end project "blog-app"
  - Use IntelliJ IDEA to open the project in a new window.
  - Enter "npm install" in terminal to install related libraries.
  - Enter "npm run build" in terminal and then enter "npm run dev" in terminal to run the project.
  - You will see a link in terminal and just click it, the web page "Leisure Town" will automatically open in the browser.



- You can use this account (account: user, password: 123) or register a new account to login to use more functions.

Future Work Plan

What should be improved in the previous 3 sprints:

1. Font changes cannot be displayed.
2. Emoji cannot be displayed in the comments.
3. Pages cannot refresh automatically after adding comments.
4. Count of comments cannot be displayed correctly.
5. Part of picture cannot be updated successfully in article detail part.
6. Refine the UI design to improve visual effects
7. Adjust the interface layout to better fit new features added in further sprints

Plan of the sprint4, 5:

1. Blogs classified by categories
2. Blogs classified by tags （front-end of 1 and 2 are completed in advance）
3. Like the blogs and display the count of likes
4. View profile
5. Edit profile
6. User can search for blogs by title
7. Trendy blogs
8. Extra PBIs to be discussed and proposed in future sprints

Reference

[1] Yun Sihe network operation and promotion, "9 age-specific website design tips," Sohu, Dec. 2016. Accessed: Apr. 2, 2022. [Online].
Available: https://www.sohu.com/a/438828227_120407639

[2] lovelydsgn, "How would you design your site for different ages," Design Front, Accessed: Apri. 4, 2022. [Online].
Available: http://www.wzsky.net/162/116439.html

Appendix

Some information about our "write articles" page editor

1. https://gitee.com/dsnull/mavonEditor/
2. https://blog.csdn.net/cnds123321/article/details/109112408