Personel blogging system

yifan Bao
Computer science Waikato University
Hamilton, Newzealand
yifanbao522@gmail.com

***Abstract*—This electronic document is demonstrating how I build this blogging system and how those parts communicate all together as a whole, also where I put the materials which I learnt from course to this blogging system.**

Keywords—component, node.js, CSS, JS, JQuery, JAVA, web, express

I. Introduction

This blogging system was made within two weeks, and I started build this blog with web-app and try to complete all the compulsory features required, then research features in web-app, then did the java GUI for administration panels, in the last day of publish the project I did the test and re-factory the code to make sure the blog is used without bug and crash. Finally the blog had the basic function like post, login, edit post with Rich text editor, edit portfolio, upload avatars and images to their own blog. For java GUI, administrator can add, check, update blog database and export/import to JSON form. However there is still a lot room for this blog system to be improved, like for user experience, web should implement animations in read more button, datetime display more friendly like show how many day before the author posted. In java GUI, could use more selectors and reduce buttons to make the user easy to use. Also this project was not documented well and that could cause hard to maintenance in future.

II. Overall Structure

A. Start project

The web was first make with git-lab project which I was granted as a Maintainer member, then I used VCS in IntelliJ to pull the initial project from git-lab, added a pre-set .gitignore file [1] under the root to filter heavy node modules.

B. Web-Application

Then I initial and installed the nodejs, in the web-app with "*npm init*", then a file called package.json is generated under the root for me to manage packages. Used handlebars to render the view, bcrypt to hash and salt the password when user sign up, use cookie-parser as middleware to handle cookie when user checked "remember me" checkbox when they sign in (but still used session to store the userinfo because run out of time for the project). Express as main route handling engine. Multer and jimp to upload image from local to host and resize it. Mariadb module to handle database connection and send DDL and DML to server. Use Nodemon to save time when debug and test my app. Use quill to implement rich text editor, readmore-js to improve user experience when browser the blog articles. Pure CSS and JS to build 2 level nested comment (post new comment only html from, so refresh was needed, could be improve) Used boostrap-5 to make the website has better view and user experience, plus some my own custom CSS to apply micro-adjustment to this blog.

In login / sign up page used same JS to check password, in signup page ajax was used to check if the username already exists in database.

C. SQL Database

Have 4 tables *userinfo* to store users details and password, *post* to store posts, *comment_reply*, *post_comment* to supply nested comments.

D. JAVA GUI namagement

Used huge amount of GUI component to support the features required. Used external libraries for additional function like Resultset, data query (mariadb Connector.jar [2]), parse database to JSON(Jackson-core.jar [3]), send email to user to reset password(email.jar [4]).

III. Materials I Used From Class

The *verifyAuthenticated* function was learn form class which is used to check if user is login by session, I also write my own one to check cookie:

```
app.use(function(req , res , next ) {
    var url = req.url;
    if (url !== '/login' && !req.signedCookies.user && url !== '/' && url!=='/resetPwd' && url!=='/fulltext') {
        res.redirect('/login')
        return
    }
    next();
});
```

Figure 1 cookie authenticate.

The basic express frame also modified from classes exercises. Like the index.js initialize the session , use res.locals to replace : content={} when use res.render("view",content), modularize routers from *Web Lab 12 : An authentication system.*

Avatar upload to portfolio and resize it was learnt from *Web Lab 16: File uploads & file processing* .

The whole DAO which was used to query data to server, learn from *Web Lab 15: Database access from node.js* , in my blog web-app I split the Dao into different files by its function, like user-dao, post-dao.

The view engine I used handlebars which was learnt from *Lab11* , I find it is powerful and timesaving. Additionally in my project I used some more handlebars function which was not practiced in that lab exercise like "../" which was use to use {{}} variable outside a nested loop, {{{}}} was used to render html from rich text editor to body.

Boostrap we learnt from *web homework 2* was used to make front-end of the site have responsive and mobile compatible ability.

GUI java we learn was used to play a very important part in design administration interface.

Table view we learnt helped me with search article based on timeframe and show / hide posts.

IV. Materials isn't From Class

A. Web-app

Quill- rich text editor with used a quill plugin to upload image in editor to imgbb.com.

B. Email sender from java

Need external library and configuration before send email, this part is easy.

C. Jquery

I find this powerful and could make code shorter, which has a lot of function that javascript do not have and Ajax function, solved the problem of username checker which blocked me for a long time. Also used to make submit without refresh the page, improved user experience.

D. Result set Table

Which I find a bit hard to understand in the beginning and convert the database to a java table model to render take me some time.

E. File choose in java

Save buttons, could use to switch statement to select different file and pre-fix.

F. Database access from java side

I learnt how to make connection to mysql server in java with import sql.* library and initialize connector, then send query to database, which has similarity with node DAO, so it was easy to learn and use.

G. Send email to reset password

I did not encrypt the link that was send to user because I did not figure out how to use JWT, and if used in real word user info could leak. Could learn JWT which is important and widely used in modern webapp.

V. Stuff I need to Improve

After the project I found there are a lot more stuff I need to learn to make the website to run like we used in our daily life, the project's structure also not well designed. Search engine in node like elasticsearch still did not figure out how it worked. For Database I should think carefully before create it, diagram didn't draw before I start this project to better overview about it. JWT token was widely used to make a verification system and password reset link which I haven't figure out how it was worked.

A. Directory design

In web app it should put all web related folder under path :
'src/main/webapp/my-app'
for node(back-end) deployment and
'src/main/webapp/my-inf'
for front-end deployment.
For JAVA gui,
'src/main/admin/gui'

B. Database Design

Database should think carefully before create it, I didn't draw any ER diagram before I created in sql file, that lead later delete all the data in that table and recreate another table to add one more column, for example, when I was doing JAVA GUI management system, a requirement that allow admin to control show and hide posts, then I have to made a new column called visible.

Also If in the future, I want to add upvote or like button for rank the comment and posts, A new table have to be recreated and that probably also cause programmer to waste a lot of time to deal with the problem of incorrect assignment when they create new table(typo).

C. Search function was not imeplmented

Slow process in deployment basic function of this webapp, so I did not get enough time to learn the search well.

D. Others

I also find the real nested thread like reddit is really hard to make, want to try that in the future.

Use the package people already made could save sometime but if programmer not have a deep understanding of how web / program works will cost them time to debug simple error. I was experience this condition when I was using their packages.

Didn't dig deep into web-side post/ user management system, didn't use a lot cookie to make better user experience, etc. set time to auto-logout . For post management like could add category for each post article, add tags, preview and draft post.

VI. Acknowledgment

I offer my sincere appreciation for the learning opportunities The University of Waikato and my lecture Sapna Jaidka, Cameron Grout, they provide a lot support to assist this project . Also my completion of the project could not have been accomplished without the support of my classmates Fiona Dunton, Jun Li, Jie Li, Xiaojuan Li, Danial Schmidt.

VII. References

[1] shiftkey , stuartpb and SimonSiefke , "gitignore/Node.gitignore," [Online]. Available: https://github.com/github/gitignore/blob/master/Node.gitignore.

[2] "mariadb," mariadb, 2021. [Online]. Available: https://mariadb.com/kb/en/about-mariadb-connector-j/.

[3] 9 January 2021. [Online]. Available: https://jar-download.com/artifacts/com.fasterxml.jackson.core.

[4] B. Shannon. [Online]. Available: https://github.com/javaee/javamail/releases/tag/JAVAMAIL-1_6_2.