Design and Implementation of a Job Portal Web Application Using Java Servlets, JSP, and MySQL

Vishal Prajapati
Dept. of MCA
GLBITM
Greater Noida, India
24150cn095@glbitm.ac.in

Dr. Lalan Kumar
Dept. of MCA
GLBITM
Greater Noida, India
lalan.kumar@glbitm.ac.in

Abstract—This paper is about the development of a web-based Job Portal built with Java Servlets, JSP and MySQL. The Job Portal runs on Apache Tomcat v8.5. The application follows an MVC architecture. It supports two authenticated roles. Admin and user. Alongside a public guest interface. Administrators can post, update and remove job listings. Registered users can. Filter those listings by location and category view details and manage their profile. Bootstrap 4 provides a front end. The paper covers architecture, module design, testing observations and a candid assessment of the limitations and planned improvements.

Keywords—job portal; Java Servlet; JSP; MySQL; MVC; JDBC; Apache Tomcat; e-recruitment

I. INTRODUCTION

Finding a job is not easy. With the internet job seekers often have to visit many websites miss relevant openings or get lost in poorly designed interfaces. On the employer side many small and mid-sized organizations still rely on channels because enterprise-grade hiring tools are either too expensive or too complex. We built a Job Portal Web Application from scratch using Java Servlets, JSP and MySQL deployed on Apache Tomcat v8.5. The idea was to keep the stack straightforward that any organization with a basic server could host it. The system supports three kinds of users. Administrators who manage the content, registered job seekers who explore and apply and guests who can browse the landing page.

This paper documents the development journey of the Job Portal Web Application. We also situate the project within the landscape of e-recruitment research to clarify where our contribution fits. The Job Portal Web Application is an easy-to-use platform for job seekers and employers.

II. RELATED WORK

Online recruitment has attracted academic attention since the early 2000s. Kumar et al. Built one of the web-based hiring systems in PHP and MySQL. We took a structural philosophy though we moved to Java EE for stronger type safety and deployment portability. Sharma and Singh added a skill-matching layer that scored candidates against open positions. We deliberately did not include a recommendation engine in this version of the Job Portal Web Application.

The MVC pattern in Java EE contexts has been studied thoroughly by Patil et al. Their analysis of maintainability metrics across servlet-based projects confirmed what most developers already sense separating presentation, logic and data makes codebases easier to debug and extend. That finding directly influenced our choices for the Job Portal Web Application. On the security side Islam et al. Catalogued vulnerabilities in portal applications and found SQL injection to

IV. MODULES & FEATURES

A. Authentication and Session Handling

Login works differently for the admin and for users. The super-admin account is authenticated by a direct string comparison in LoginServlet. Regular users go through UserDao.Login() which queries the database with a PreparedStatement. On success the user object is stored in the session.

B. Job Posting and Administration

The admin dashboard lives in admin.jsp. Is the only page from which job management is possible. Posting a job hits AddPostServlet, which reads five fields and passes them to jobDAO.addjob(). Editing goes through UpdateJobServlet; deletion through DeleteServlet.

C. Job Search and Filtering

The home.jsp page shows all listings when a user first loads it. They can narrow results using a form with two dropdowns. Location and category. Which posts to more_view.jsp. Two DAO methods handle the filtering: getjobsANDlocationAndCate() for cases where both filters are set and getjobsORlocationAndCate() when only one is selected.

D. User Registration and Profile

Registration is handled by RegisterServlet, which calls UserDao.addUser() after reading the form fields. Profile editing goes through edit_profile.jsp and UpdateProfileServlet, which calls UserDao.updateUser().

E. Landing Page

The index.jsp public page is meant to give first-time. The index.jsp public page is meant to give first-time visitors a sense of what the Job Portal Web Application offers before they commit to registering. It includes a Bootstrap carousel with images a category grid showing the job sectors the portal covers, a testimonials section and a footer.

V. TESTING AND RESULTS

We tested the Job Portal Web Application manually against a MySQL instance running through all the main user flows: registration, login as admin and as user posting a job searching by category filtering by location editing a profile and logging out. Everything behaved as expected. The PreparedStatement approach held up against our injection test cases.

In terms of scale the codebase ended up at 6 servlet controllers 2 DAO classes, 2 entity classes, 12 JSP pages and DBconnect.java as the shared connection utility. Page load times for the job listing page stayed under 200ms on a standard development laptop.

The clearest limitation is password storage. Passwords are saved in text, which is indefensible in a production setting. We also noticed that the admin credential hardcode in LoginServlet is a point of failure. Both issues need to be fixed before the Job Portal Web Application could go live

VI. CONCLUSION

be the most prevalent. We addressed this throughout by using PreparedStatement never concatenating user input into query strings.

What prior work shares is a reliance on either proprietary frameworks or languages other than Java. Our system, the Job Portal Web Application is intentionally built on vanilla Java EE making it easy to understand modify and deploy without a learning curve.

III. SYSTEM ARCHITECTURE

A. Technology Stack

The Job Portal Web Application is organized into three tiers. The presentation tier uses JSP pages styled with Bootstrap 4. The business tier is entirely Java Servlets, which handle request routing, session management and interaction with the DAO layer. The data tier is MySQL 8.0 accessed through JDBC using PreparedStatement objects.

Component	Technology Used
Frontend	JSP + Bootstrap 4
Backend	Java Servlets (Java EE)
Database	MySQL 8.0
Server	Apache Tomcat v8.5
DB Access	JDBC PreparedStatement
Build Tool	Maven
IDE	Eclipse IDE

TABLE I. Technology Stack of the Job Portal

B. MVC Design Pattern

We followed MVC strictly. The entity classes hold data and nothing else. The DAO classes contain all the SQL. The servlets sit in the middle: they receive a request call whichever DAO method they need attach the results to the request or session and forward to the JSP. The JSPs render output using JSTL and occasional scriplets.

One practical consequence of this separation was that when we needed to add the job filtering feature in development the change was isolated to one DAO method and one JSP. The servlets barely changed.

C. Database Schema

The database has two tables. The user table stores id, name, qualification, email, password and role. The jobs table stores id, title, description, category, status, location and date. The status column drives visibility. Records with status set to 'Active' appear on the user-facing listing page.

Building the Job Portal Web Application gave us an understanding of how Java EE components fit together in a real application. The MVC approach paid off the JDBC layer was straightforward to write, and Bootstrap handled the layout without much fuss. The Job Portal Web Application is an easy-to-use platform, for job seekers and employers.

There is still a lot of work to be done on this project. We need to do password hashing with BCrypt, store admin credentials in the database set up email verification when people sign up create a feature to upload resumes and eventually come up with a way to recommend jobs based on the qualifications that users have listed. These are the next steps to take. The code is set up in a way that makes it easy to add these features one by one than having to start all over again. This seems like a place to stop for now.

ACKNOWLEDGMENT

We want to thank the faculty of the Department of MCA at GLBITM in Greater Noida for helping us and giving us feedback on this project.

REFERENCES

[1] S. Kumar, A. Verma and P. Gupta wrote a paper called "Online Recruitment System Using PHP and MySQL" in the International Journal of Computer Applications in 2015. This paper is very relevant to our project because it talks about how to build a system for recruiting people

[2] R. Sharma and M. Singh gave a presentation at the International Conference on Computational Intelligence in 2018 about recommending jobs based on skills in e-recruitment systems. Their presentation is important to our project because it discusses how to match people with jobs that fit their skills. This paper is helpful to our project because it explains how to use the MVC architecture in Java web development.

[3] M. Islam, T. Ahmed and H. Kabir wrote a paper called "Security Analysis of Web-Based Portals". Presented it at the IEEE International Conference on Cyber Security in 2020. This paper is important to our project because it talks about how to keep web-based portals secure.

[4] J. Hunter and W. Crawford wrote a book called Java Servlet Programming that was published by O'Reilly Media in 2001. This book is helpful to our project because it explains how to program Java servlets.

[5] B. Basham, K. Sierra and B. Bates wrote a book called Head First Servlets and JSP that was published by O'Reilly Media in 2008.