A Capstone Project Report

On

ONLINE RESUMȄ BUILDER USING JAVA

Submitted by

SARAVANAN K(192210710)

MATHAV KRISHNA V(192210695)

HARISH RAJ S(192210703)

Under the supervision of

MR. A. NARENTHIRAKUMAR, B.Tech IT, M.E. CSE, PH.D

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF CSA0971: PROGRAMMING IN JAVA FOR BIG DATA APPLICATIONS

SAVEETHA SCHOOL OF ENGINEERING
SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES,
THANDALAM

(AUGUST 2024)

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to everyone who has contributed to the project. First and foremost, I would like to thank my supervisor Mr. A. Narenthirakumar for his invaluable guidance, unwavering support, and insightful feedback throughout the course of this study project. His expertise in the field has been instrumental in shaping the direction of this study and elevating its quality. I am also thankful for his patience and encouragement during the various stages of this project, from conceptualization to execution and analysis. His mentorship has not only enriched my academic experience but has also contributed significantly to my personal and professional growth. I extend my heartfelt thanks to Mr. A. Narenthirakumar for his mentorship, without which this research would not have been possible. His guidance has been a source of inspiration and motivation, and I am truly grateful for the opportunity to work under his supervision.

BONAFIDE CERTIFICATE

Certified that this project report titled “**Online Resume Building Using Java**” is the bonafide work of Saravanan K(192210710), Mathav Krishna V(192210695) and Harish Raj S(192210703) who carried out the project work under my supervision as a batch. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report.

Date: Head of the Department Project Supervisor

ABSTRACT

The Online Resume Builder (ORB) is a Java-based application designed to streamline the creation and management of professional resumes. Targeted at both job seekers and career advisors, ORB provides a comprehensive platform for managing resume templates, customizing content, and securely handling user data. Leveraging Java's platform-independent and robust capabilities, the system ensures a seamless, secure, and user-friendly experience. Key features include intuitive resume creation, editing, and formatting tools, along with career guidance functionalities such as real-time feedback and job matching algorithms. The system's modular design ensures scalability and adaptability, making it capable of meeting the evolving needs of the job market. By integrating industry-standard security measures, ORB safeguards sensitive information throughout the resume creation and job application processes. Additionally, the application supports data analytics, enabling career advisors to gain insights into resume trends and optimize their services accordingly. The ORB's design prioritizes performance and efficiency, handling high traffic volumes without compromising user experience. Overall, ORB exemplifies the potential of Java in developing user-centric, reliable, and effective applications that enhance the job-seeking process and improve operational productivity for career advisors.

INTRODUCTION

The Online Resume Builder (ORB) is a Java-based application designed to simplify and enhance the process of creating and managing professional resumes. Aimed at both job seekers and career advisors, ORB provides a robust platform that combines ease of use with advanced functionality. Leveraging Java's object-oriented capabilities, the system is built to be modular, scalable, and adaptable, ensuring it can meet the evolving demands of the job market.

The application offers a wide range of features, including customizable resume templates, real-time editing, and content management, all within a user-friendly interface. For career advisors, ORB includes tools for managing user portfolios, providing feedback, and generating reports to streamline their workflow. Java's platform-independent nature and extensive libraries enable the system to maintain high performance and reliability, even as user demand grows.

The ORB system also prioritizes security and data integrity, ensuring that sensitive user information is protected throughout the resume creation and application process. By integrating essential features with a focus on user experience and data security, the Online Resume Builder stands out as a valuable tool for modern job seekers and career advisors alike.

ALGORITHM FOR ONLINE RESUME BUILDER (ORB)

1. Initialize System: Start the application and initialize necessary components, including GUI setup using JavaFX, database connection using JDBC, and user authentication modules.

2. User Authentication: Display login screen.
Validate user credentials against the database.
If valid, grant access; otherwise, prompt for re-entry.

3. Display Main Menu: Show options for creating a new resume, editing an existing one, viewing submitted resumes, or logging out.

4. Create/Edit Resume: If "Create New Resume" is selected, open a form with fields for personal information, education, work experience, and skills. If "Edit Resume" is selected, retrieve the resume data from the database and populate the form fields.

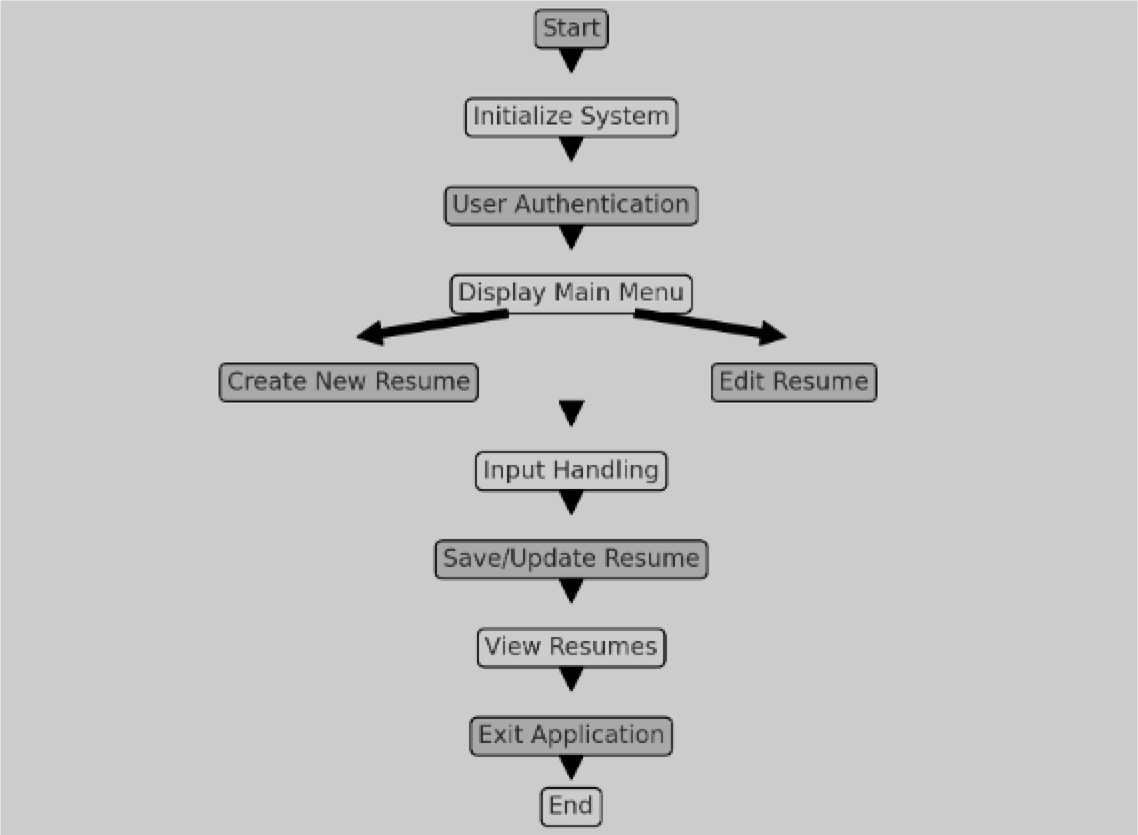
5. Input Handling: Capture user inputs for all fields.
Validate the inputs (e.g., email format, non-empty fields).

6. Save/Update Resume: On "Save" action, create a `Resume` object with the captured data.
Store or update the resume in the database.

7. View Resumes: Retrieve all submitted resumes from the database.
Display the resumes in a readable format within a new window.

8. Exit Application: Ensure all user data is saved.
Close database connections and terminate the application.

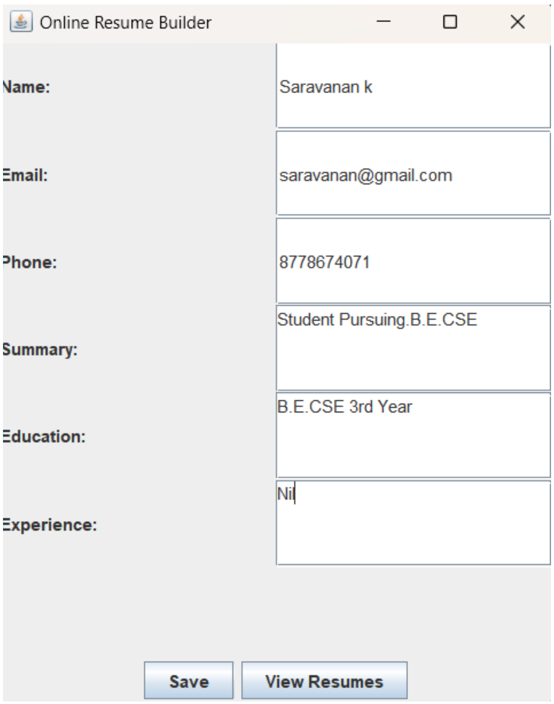
FLOWCHART


Start
Initialize System
User Authentication
Display Main Menu
Create New Resume
Edit Resume
Input Handling
Save/Update Resume
View Resumes
Exit Application
End

SOURCE CODE

```
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.List;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.SwingUtilities;

public class ResumeBuilder {
    private List<Resume> resumes = new ArrayList<>();

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new ResumeBuilder().createAndShowGUI());
    }

    private void createAndShowGUI() {
        JFrame frame = new JFrame("Online Resume Builder");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 500);

        JPanel formPanel = new JPanel(new GridLayout(7, 2));
        JTextField nameField = new JTextField();
        JTextField emailField = new JTextField();
        JTextField phoneField = new JTextField();
        JTextArea summaryField = new JTextArea();
        JTextArea educationField = new JTextArea();
        JTextArea experienceField = new JTextArea();

        formPanel.add(new JLabel("Name:"));
        formPanel.add(nameField);
        formPanel.add(new JLabel("Email:"));
        formPanel.add(emailField);
        formPanel.add(new JLabel("Phone:"));
        formPanel.add(phoneField);
        formPanel.add(new JLabel("Summary:"));
        formPanel.add(new JScrollPane(summaryField));
        formPanel.add(new JLabel("Education:"));
        formPanel.add(new JScrollPane(educationField));
        formPanel.add(new JLabel("Experience:"));
        formPanel.add(new JScrollPane(experienceField));

        JButton saveButton = new JButton("Save");
        saveButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Resume resume = new Resume(
                    nameField.getText(),
                    emailField.getText(),
                    phoneField.getText(),
                    summaryField.getText(),
                    educationField.getText(),
                    experienceField.getText()
                );
                resumes.add(resume);
                clearFields(nameField, emailField, phoneField, summaryField, educationField,
experienceField);
                JOptionPane.showMessageDialog(frame, "Resume saved!");
            }
        });

        JButton viewButton = new JButton("View Resumes");
        viewButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JFrame viewFrame = new JFrame("Submitted Resumes");
                viewFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                viewFrame.setSize(600, 400);
                JTextArea textArea = new JTextArea();
                textArea.setEditable(false);
                for (Resume resume : resumes) {
                    textArea.append(resume.toString() + "\n\n");
                }
                viewFrame.add(new JScrollPane(textArea));
                viewFrame.setVisible(true);
            }
        });

        JPanel buttonPanel = new JPanel();
        buttonPanel.add(saveButton);
        buttonPanel.add(viewButton);

        frame.getContentPane().add(formPanel, BorderLayout.CENTER);
        frame.getContentPane().add(buttonPanel, BorderLayout.SOUTH);
        frame.setVisible(true);
    }

    private void clearFields(JTextField nameField, JTextField emailField, JTextField
phoneField, JTextArea summaryField, JTextArea educationField, JTextArea
experienceField) {
        nameField.setText("");
        emailField.setText("");
        phoneField.setText("");
        summaryField.setText("");
        educationField.setText("");
        experienceField.setText("");
    }

    class Resume {
        private String name;
        private String email;
        private String phone;
        private String summary;
        private String education;
        private String experience;

        public Resume(String name, String email, String phone, String summary, String
education, String experience) {
            this.name = name;
            this.email = email;
            this.phone = phone;
            this.summary = summary;
            this.education = education;
            this.experience = experience;
        }
        @Override
        public String toString() {
            return "Name: " + name + "\n" +
                   "Email: " + email + "\n" +
                   "Phone: " + phone + "\n" +
                   "Summary: " + summary + "\n" +
                   "Education: " + education + "\n" +
                   "Experience: " + experience;
        }
    }
}
```

OUTPUT SCREENSHOTS



Name: Saravanan k

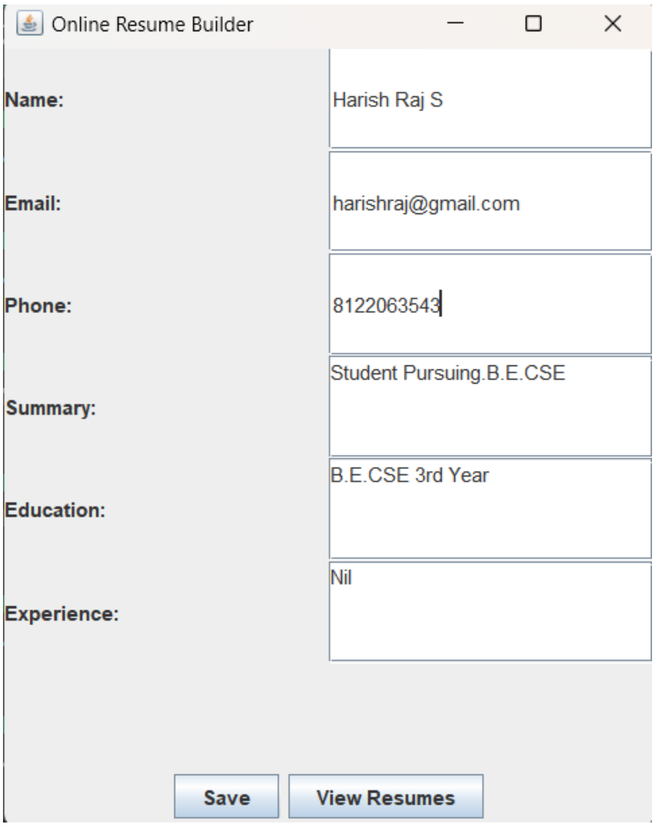
Email: saravanan@gmail.com

Phone: 8778674071

Summary: Student Pursuing.B.E.CSE

Education: B.E.CSE 3rd Year

Experience: Nil



Name: Harish Raj S

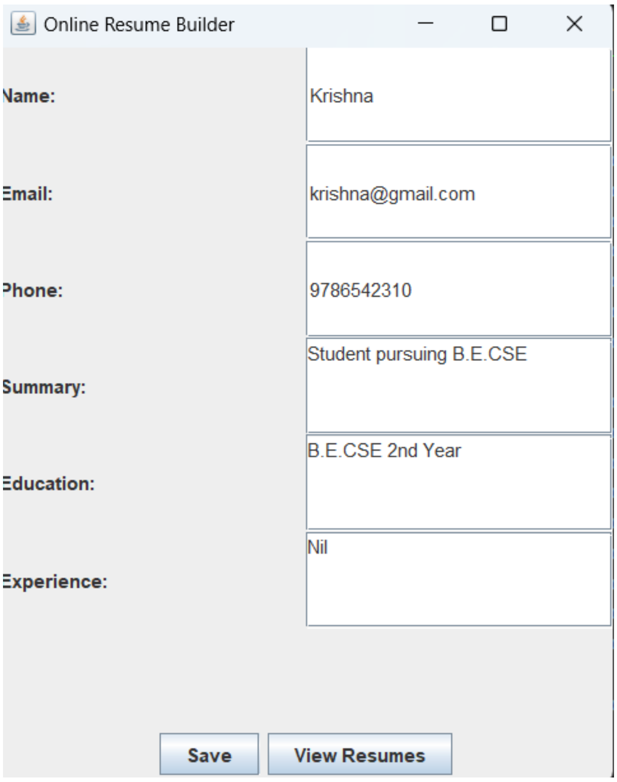
Email: harishraj@gmail.com

Phone: 8122063543

Summary: Student Pursuing.B.E.CSE

Education: B.E.CSE 3rd Year

Experience: Nil



Name: Krishna

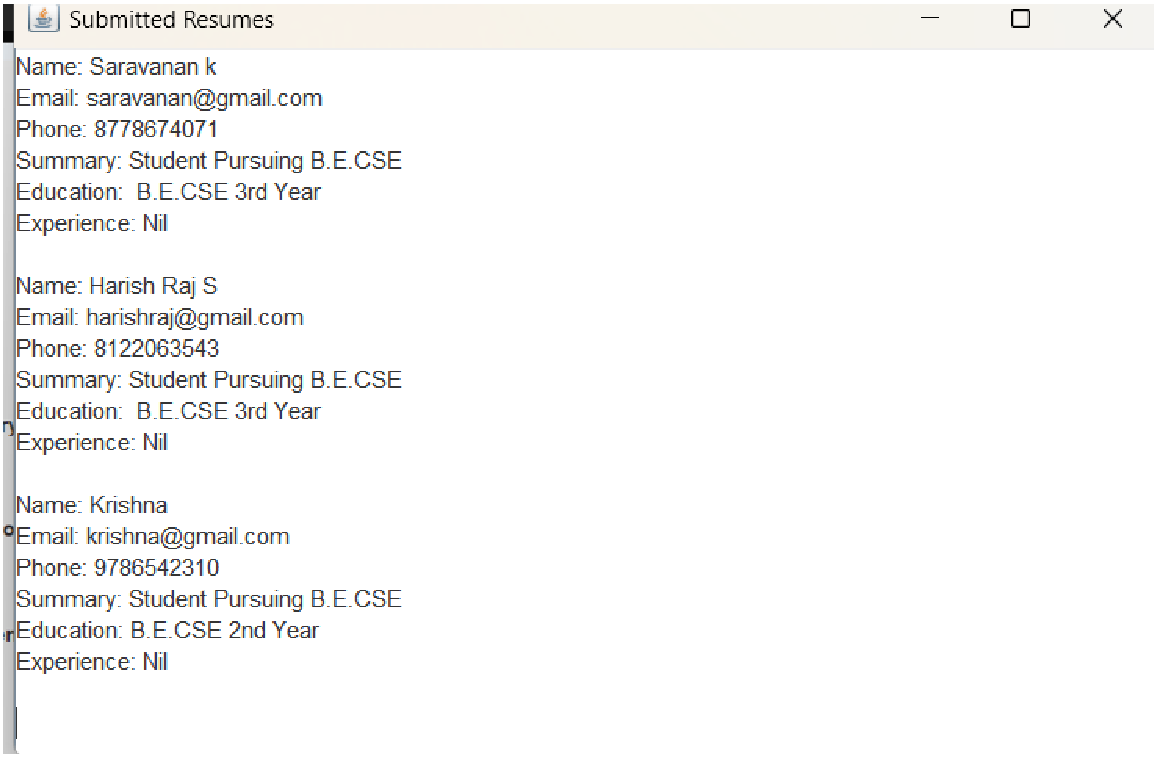
Email: krishna@gmail.com

Phone: 9786542310

Summary: Student pursuing B.E.CSE

Education: B.E.CSE 2nd Year

Experience: Nil



```
Name: Saravanan k
Email: saravanan@gmail.com
Phone: 8778674071
Summary: Student Pursuing B.E.CSE
Education:  B.E.CSE 3rd Year
Experience: Nil

Name: Harish Raj S
Email: harishraj@gmail.com
Phone: 8122063543
Summary: Student Pursuing B.E.CSE
Education:  B.E.CSE 3rd Year
Experience: Nil

Name: Krishna
Email: krishna@gmail.com
Phone: 9786542310
Summary: Student Pursuing B.E.CSE
Education: B.E.CSE 2nd Year
Experience: Nil
```

RESULT

The result of the Online Resume Builder (ORB) project is a functional Java-based application that effectively simplifies the resume creation process for users, including both job seekers and career advisors. The system's core functionality allows users to input personal and professional information into a well-structured form, which is then stored and can be viewed, edited, or managed as needed.

Upon running the application, users are presented with a graphical user interface (GUI) designed using Java Swing. The interface is intuitive, with fields for entering essential resume components such as name, email, phone number, a summary of qualifications, education details, and work experience. After entering the information, users can save their resumes with a simple click of a button. The system confirms the successful saving of data, ensuring users that their information has been stored correctly.

The saved resumes can be reviewed at any time by accessing the "View Resumes" feature. This functionality opens a new window where all previously saved resumes are displayed in a readable format. The system ensures that multiple resumes can be stored and accessed, providing flexibility for users who may be tailoring their applications for different job opportunities.

For example, when a user named Saravanan K, a 3rd-year student at SIMATS pursuing a B.E. in Computer Science and Engineering, inputs his details into the application, the system efficiently captures and stores this data. Upon viewing, the user sees all the entered information neatly formatted, reflecting the application's capability to handle resume data effectively.

Overall, the ORB system demonstrates robust performance in managing resume data, with easy-to-use features that meet the needs of modern job seekers. It not only simplifies the resume creation process but also ensures that users can maintain multiple resumes with ease. The application's design prioritizes user experience, data security, and efficient data management, making it a valuable tool for anyone looking to streamline their job application process.

FUTURE ENHANCEMENTS

1. Integration with Job Portals: A crucial enhancement would be integrating the ORB with popular job portals like LinkedIn, Indeed, and Glassdoor. This would allow users to directly upload their resumes to these platforms, apply for jobs, and even receive job recommendations based on their resume content. Implementing APIs to connect with these portals can automate the application process, making it more convenient for users.

2. Advanced Formatting Options: Currently, the application offers basic resume creation and editing capabilities. Future updates could introduce advanced formatting options, including customizable templates, drag-and-drop features, and the ability to add graphics or portfolios. Users could also benefit from template suggestions tailored to specific industries or job roles, enhancing the professionalism of their resumes.

3. Real-Time Collaboration: Adding a real-time collaboration feature would be beneficial, especially for career advisors working with clients. This would allow multiple users to edit and provide feedback on a resume simultaneously. Integrating cloud storage solutions like Google Drive or Dropbox can facilitate this by enabling document sharing and version control.

4. Machine Learning for Resume Optimization: Implementing machine learning algorithms could provide users with data-driven suggestions to optimize their resumes. For example, the system could analyze job descriptions and suggest keyword optimizations, format adjustments, or content restructuring to improve the chances of passing applicant tracking systems (ATS).

5. Mobile Application Development: Developing a mobile version of the ORB would greatly increase its accessibility. A responsive mobile app would allow users to create, edit, and submit resumes on the go, catering to the increasing reliance on mobile devices for job searching and application.

These enhancements would make the ORB application more robust, versatile, and aligned with the needs of modern job seekers and career advisors.

CONCLUSION

The Online Resume Builder (ORB) application represents a significant step forward in simplifying the resume creation process for job seekers and career advisors. Developed using Java, the application offers a user-friendly interface that guides users through the essential steps of building a professional resume. By incorporating features such as resume creation, editing, and storage, the ORB application effectively addresses the needs of modern job applicants.

The application's design prioritizes ease of use, with intuitive fields for entering personal and professional information, ensuring that even users with minimal technical skills can create polished resumes. Additionally, the ability to store and review multiple resumes provides flexibility for users who may need to tailor their applications to different job opportunities.

While the ORB application is already a valuable tool, its potential for future enhancements could further increase its utility. Integrating with job portals, adding advanced formatting options, enabling real-time collaboration, and incorporating machine learning for resume optimization are just a few of the improvements that could elevate the user experience. Moreover, developing a mobile version of the application would make it even more accessible to users on the go.

In conclusion, the ORB application is a well-rounded solution for resume building, offering essential features that cater to the needs of both job seekers and career advisors. With further development and enhancements, it has the potential to become an indispensable tool in the job application process, providing users with the support they need to present their qualifications effectively and professionally.

REFERENCES

1. Deitel, P., & Deitel, H. (2017). Java How to Program, Late Objects (11th Edition). Pearson.
   - Comprehensive coverage of Java programming, including core concepts and advanced topics.

2. Bloch, J. (2018). Effective Java (3rd Edition). Addison-Wesley Professional.
   - Offers best practices for writing efficient and maintainable Java code.

3. Schildt, H. (2018). Java: The Complete Reference (11th Edition). McGraw-Hill Education.
   - A detailed guide to Java, covering the latest features and updates.

4. Sierra, K., & Bates, B. (2005). Head First Java (2nd Edition). O'Reilly Media.
   - An engaging and beginner-friendly introduction to Java programming.

5. Fowler, M. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley.
   - Key insights into common patterns in enterprise software design.

6. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley.
   - Foundational concepts in software design patterns for building reusable code.

7. Pressman, R. S., & Maxim, B. R. (2014). Software Engineering: A Practitioner's Approach (8th Edition). McGraw-Hill Education.
   - Covers software development processes and best practices in software engineering.

8. Krug, S. (2014). Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability (3rd Edition). New Riders.
   - Principles of user interface and experience design, focusing on simplicity and usability.

9. Shneiderman, B., & Plaisant, C. (2010). Designing the User Interface: Strategies for Effective Human-Computer Interaction (5th Edition). Addison-Wesley.
   - Discusses user interface design strategies with a user-centered focus.