Programming Enterprise Components

Table of Contents

[Introduction 4](#_Toc100342292)

[Brief Introduction of Data Visualisation 4](#_Toc100342293)

[Visualisation – 1 5](#_Toc100342294)

[Visualisation – 2 6](#_Toc100342295)

[Visualisation – 3 7](#_Toc100342296)

[Visualisation – 4 8](#_Toc100342297)

[Visualisation – 5 9](#_Toc100342298)

[Visualisation – 6 10](#_Toc100342299)

[Visualisation – 7 11](#_Toc100342300)

[Visualisation – 8 12](#_Toc100342301)

[Critical Review 13](#_Toc100342302)

[Conclusion 13](#_Toc100342303)

[References 14](#_Toc100342304)

# List of Figures

[Figure 1: Visualisation – 1 (Bar Chart) 5](#_Toc100342385)

[Figure 2: Visualisation – 2 (Histogram) 6](#_Toc100342386)

[Figure 3: Visualisation - 3 (Time Series - 1) 7](#_Toc100342387)

[Figure 4: Visualisation - 3 (Time Series - 2) 7](#_Toc100342388)

[Figure 5: Visualisation - 4 (Autocorrelation) 8](#_Toc100342389)

[Figure 6: Visualisation - 5 (Box Plot) 9](#_Toc100342390)

[Figure 7: Visualisation - 6 (Radar Plot) 10](#_Toc100342391)

[Figure 8: Visualisation - 7 (Heat Map) 11](#_Toc100342392)

[Figure 9: Visualisation - 8 (Bubble Chart) 12](#_Toc100342393)

| **Your name** | **Priyank Hetalkumar Patel** | **Your Student ID** | **001184935** |
| --- | --- | --- | --- |

|  |  |  |  |
| --- | --- | --- | --- |
| **Other members in the Group:** | **Arpit Kumar Singh** | **Their IDs** | **001201546** |
| **Deepanshu Upadhyay** | **001182406** |
| **Prasanna Rahul Madugula** | **001192128** |
| **Trishant Sharma** | **001199934** |

# Brief statement of features you have implemented

|  |  |  |
| --- | --- | --- |
| **Feature** | **Status** | **Your Comments** |
| **Functionality A** | Fully completed  Partially completed  Having bugs/Not working  Not implemented | MySQL database is used and all the tables are fully normalise. All the constraints and relationships are well managed in the database. For application Jakarta EE is used with session beans and JSP is used for the UI part. |
| **Functionality B** | Fully completed  Partially completed  Having bugs/Not working  Not implemented | Employees can log in to the system and able to fill the holiday request form. Also they can view the list of their own holiday requests. |
| **Functionality C** | Fully completed  Partially completed  Having bugs/Not working  Not implemented | Admin user can view list of pending requests and react accordingly by checking the constraints. Even admin can filter list according to the date and employees. |
| **Functionality D** | Fully completed  Partially completed  Having bugs/Not working  Not implemented | All the constraints are checked automatically as soon as employee submit the request and admin can see the breaking constraints in the request list. |
| **Functionality E** | Fully completed  Partially completed  Having bugs/Not working  Not implemented | Rest APIs are created for login and submit holiday request form. |
| **Functionality F** | Fully completed  Partially completed  Having bugs/Not working  Not implemented | Message driven beans are created to get the alert when any request is sent via web service and this alert will be stored in the database. |
| **Functionality G** | Fully completed  Partially completed  Having bugs/Not working  Not implemented | Merge sort algorithm is used to sort holiday request and prioritise them according to the previous data and constraints. |

# PART - A

## Design Documentation:

Graphical user interface, diagram

Description automatically generated

Figure 1: Architecture of Holiday Booking System

Diagram

Description automatically generated

Figure 2: Entity Relationship Diagram

The visual representation of different entities involved in the system and their relation to each other is represented in the ERD. Each employee is a person and has the attributes of a person. The Person has the attributes of the Name, Address, Postcode, Date of Birth, and PhoneNumber. The Employee entity has other attributes ID, Password, Email, Department, Role, JoinDate, and Number of Leaves. The LeaveLogs entity is related to the employee and has attributes ID, LeavesLeft, and Date which are multi-valued as leaves can be multiple dates. The HolidayRequest is made by the employee and has the attributes ID, Status, To date, From date, Number of Days, and Reason. The Holiday request is a multi-valued entity as an employee can make multiple holiday requests. The Alert is an entity with attributes ID, Title, Message, and Viewed.

## Screenshots:

A picture containing text, indoor, wall

Description automatically generated

Figure 3: Login Page

A screenshot of a computer

Description automatically generated

Figure 4: Admin Dashboard

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Figure 5: Create new employee Form

A screenshot of a computer

Description automatically generated with medium confidence

Figure 6: Employee list of Straight Walls Ltd.

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Figure 7: Update employee form

Text

Description automatically generated with low confidence

Figure 8: Employee dashboard

Graphical user interface

Description automatically generated with low confidence

Figure 9: Employee holiday request form

A screenshot of a computer

Description automatically generated with medium confidence

Figure 10: Holiday requests list of employees

Graphical user interface, text, website

Description automatically generated

Figure 11: Employee list with their working status

Graphical user interface

Description automatically generated

Figure 12: Leave requests

Graphical user interface, text

Description automatically generated

Figure 13: List of pending leave request to be responded

Graphical user interface, text, application, email

Description automatically generated

Figure 14: Alert messages saved in the database when any request generated

Text

Description automatically generated

Figure 15: Code for the message beans

A screenshot of a computer

Description automatically generated with medium confidence

Figure 16: API for the employee login

A screenshot of a computer

Description automatically generated with medium confidenceFigure 17: API for the leave request form

## Evaluation:

First, we have been approached by the company Straight Walls Ltd. It wants us to develop an automated holiday booking system. The requirements are given like in the system admin can be able to add employees and perform edit and delete operations on it. Employees can request the holidays and the admin can approve or reject that request but for that, there are some constraints which need to be followed and the system should check those constraints automatically. Even there are different departments in the company and each department has different roles for the employees.

So, We did requirement analysis and divided our tasks according to the requirement in the initial phase. There are some limitations to the technologies and environments for the development which we kept in mind during the implementation of the holiday booking system. For the first phase of implementation, we planned to work on the database and some basic functionalities like CRED of employees and log in. So, we designed Entity Relationship Diagram and created a database in MySQL from that design and started working on the Jakarta EE environment and set up the application architecture and separate client web and EJB module which include business logic. And clients communicate through the interfaces which are stored in the “remoteDAO” file.

After the setup, we started working on functionality implementations which we have divided into groups so that everyone can work on their modules and we can avoid the interdependency of every member in the group. Though we face some issues working in a group as every member of the group was not available at the same time, so we won’t be able to discuss everything with everyone. We overcome that issue by using the git version controlling system and generating tasks for each group member and assigning them so that everyone knows what they have to work on and everyone can work on their branch and then we merge everyone’s work in the main branch at the end. There are some of the challenging functionalities which we have implemented like constraints check and sorting the holiday request list according to the previously taken holidays. These complex tasks take most of the time for our implementation.

There is one Web application we have created which leads to the login page at the compilation of the application. There is a session timeout added for the 15min to the application server which will be updated at the time of login. There is role-based access is added for the user. If the logged-in user is an admin then they can add employees, edit employees and delete employees with they can see some lists like employee list, holiday request list. All the links are added to the dashboard which can be accessed after login. If the logged-in user is an employee other than the admin then also they will land up on the dashboard but they can see different links like holiday request form, list of own holiday requests. Rest APIs are also compiled through the same application and they can be accessed using the same URL on which the web application is running. We have tested it using the Postman tool which is preferred for the testing of Rest API.

We have faced some issues within the group during the evolution of the application process as everyone has different background and knowledge and some have no background in coding and handling that situation and managing the whole group together and getting the same amount of work from each member is a difficult task still we managed to finish the whole project with the whole implementation of each task and all the functionalities are implemented with the perfect logics and code efficiency which are the key aspects of the development.

# PART – B

## Research

## Individual Reflection

# Coursework Contribution (completed as a group)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Team member name** | **Student ID** | **individual overall work contribution (%)** | **Signature** | **Note** |
| Student: Priyank Hetalkumar Patel | 001184935 | 30% | Priyank Patel |  |
| Student: Arpit Kumar Singh | 001201546 | 40% | Arpit Singh |  |
| Student: Deepanshu Upadhyay | 001182406 | 10% | Deepanshu Upadhyay |  |
| Student: Prasanna Rahul Madugula | 001192128 | 10% | Rahul |  |
| Student: Trishant Sharma | 001199934 | 10% | Trishant Sharma |  |
| **Total 100%** | | |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Describe each task you performed | Student: 1  work contribution in % | Student: 2  work contribution in % | Student: 3  work contribution in % | Student: 4  work contribution in % | Student: 5  work contribution in % | **total** |
| Feature A: | 50% | 50% | 0% | 0% | 0% | 100% |
| Feature B: | 20% | 20% | 20% | 20% | 20% | 100% |
| Feature C: | 10% | 10% | 30% | 30% | 20% | 100% |
| Feature D: | 40% | 60% | 0% | 0% | 0% | 100% |
| Feature E: | 0% | 0% | 40% | 30% | 30% | 100% |
| Feature F: | 30% | 70% | 0% | 0% | 0% | 100% |
| Feature G: | 70% | 30% | 0% | 0% | 0% | 100% |

# References

Eclipse-ee4j.github.io. 2019. *Overview of Enterprise Applications*. [online] Available at: <https://eclipse-ee4j.github.io/jakartaee-firstcup/jakarta-ee001.html> [Accessed 26 April 2022].

Existek.com. 2020. *Node.js vs Java in Enterprise Web App Development - Existek Blog*. [online] Available at: <https://existek.com/blog/node-js-vs-java-enterprise-web-applications-development/> [Accessed 26 April 2022].

www.javatpoint.com. 2022. *Java EE vs Node.js - Javatpoint*. [online] Available at: <https://www.javatpoint.com/java-ee-vs-nodejs> [Accessed 26 April 2022].