Bilkent University

Department of Computer Engineering

Senior Design Project

Interly.ai

Detailed Design Report

Arda Serim	21802554
Erhan Er	21801809
Furkan Turunç	21802197
Murat Furkan Uğurlu	21802062
Osman Semih Tiryaki	21801994

Supervisor: Hamdi Dibeklioğlu

Jury Members: Erhan Dolak, Tağmaç Topal, Hamdi Dibeklioğlu

Mar 13, 2023

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

1. Introduction

With the entrance of Covid19 into our lives, a lot of companies switched to online interviews [1] and even though the outbreak has diminished, online interviews are still in use for companies [2]. There are several advantages of online meetings such as easy access between the interviewee and the interviewer, or a faster recruitment process. However, there are also some disadvantageous sides of online meetings. Since the meeting is not as strong as a face-to-face interview, interviewers might not analyze the personality of the interviewees well enough to make sure that s/he is the right person to employ.

1.1 Purpose of the system

Interly.ai aims to transform the way of interviews bringing a new dimension to online meetings. It will help human resources see the characteristic features of the applicants by providing a live video conference environment and analyzing their facial expressions, and gestures using visual data. The interviewee's personality, current mood, level of anxiety, and self-confidence are presented to the interviewer's screen real-timely.

1.2 Design goals

1.2.1 Reliability

- The application should give analysis data of the interviewee with the highest possible accuracy.

1.2.2 Privacy

- Interviewee's analysis will be stored in a database to make it available to the interviewer should they want to see it again in the future. Hence, this data should be secured by hashing the data.
- Face data of interviewees will be processed by the application. This data should be processed on interviewees' computers before uploading it to the server in order to prevent uploading pure face data.

1.2.3 Efficiency

- The application should analyze the personality of the interviewee in real time. Hence, most of the processing should be done at the server side to be able to make it possible.

1.2.4 Usability

- The application should be easy to use for both interviewee and interviewer. It should be seen by users as a very basic meeting application without complex functionalities to use.
- The application should give analysis data of interviewees with an understandable and easy to read GUI to the interviewers.

1.2.5 Scalability

- As the number of users increase, the more companies utilize Interly, more data will be sent to servers and their final process will happen here. Thus, workload on central servers will increase drastically. Servers should be able to handle these kinds of situations.
- The application should keep the delay as little as possible even when more users are on the calls and should try to provide real-time analysis all the time.

1.3 Overview

In this report, an analysis of the system is explained under the current software architecture and proposed software architecture sections. In the current software architecture section, what is on the market and how Interly.ai will bring a new breath of fresh air to the market are discussed. In the proposed software architecture section, how Interly.ai will function, what will be the responsibilities of subsystems, hardware/software mapping, data management, access control and system models are explained. Then subsystems will be explained in detail. Then the test cases for ensuring that the system achieves its design goals are listed.

2. Current software architecture

Companies use existing video conferencing applications in their recruitment processes, and the analysis of the interviewees is done by the interviewers' own human vision and perceptions. Interviewers examine the behavior of the candidate by considering the candidate’s responses to the questions. They sometimes take notes for the candidate’s analysis.

There are some software solutions that use AI to help recruiters through the recruitment process. One of them is VireUp, an AI based interview application that provides analysis of applicants’ answers to the questions with graphs and heatmaps [3]. Another one is easyhire which is a software solution that analyzes only the emotions of the applicants using AI [4].

Interly.ai will bring a new innovation to the market through enabling real-time behavioral analysis of the candidate during video conference. This innovation is an incremental service innovation because it brings new facilities to an ordinary video-conferencing application in order to better evaluate the candidates during the interview. Hence computer vision technology will replace human vision and perception in the interview analysis through Interly.ai.

3. Proposed software architecture

3.1 Overview

Interly.ai is a web-based online meeting application that companies can utilize in their interview processes with the help of behavioral analysis of the interviewees. Companies’ HR managers and interviewers can create an interview session for the candidates in their hiring processes and send the online interview link to them via email. An interviewer does not have to agree to behavioral analysis on himself/herself. In such a case, it is at the discretion of the interviewer whether to continue the interview without analysis. If the interviewer accepts to keep going on

with the interview without analysis, the meeting gets started and finished after the session ends. Otherwise, the interviewer can opt to cancel the meeting.

If the interviewee agrees to behavioral analysis as expected, the meeting gets started by the interviewer, the behavioral analysis is done in real time and all the data collected and processed in the interviewee's local environment are sent to the interviewer's side. The analysis data sent to the interviewer's side is projected on the screen with easy-to-understand user interfaces and displayed instantly on graphs. After the interview session ends, the overall and instant results are saved to the Data Hangar, which is a part of the webpage where interviewers can see the interviewees' personal information and analysis results.

Moreover, the HR managers can see the statistics about what types of characteristics the employed interviewees have, and how their behaviors and moods were during their interviews. Therefore, the managers can see the details of employment and analyze the performance of interviewers about how well they eliminated and hired the candidate employees.

3.2 Subsystem decomposition

Interly.ai is a web based program that utilizes desktop plugins to cooperate simultaneously with the video conferencing application. This infrastructure is achieved using a modified client-server architecture in its fundamental structure. There are four main subsystems that enable Interly.ai to offer its users proposed functionality.

Interviewee Client is the subsystem that operates on the interviewee's machine. This part of the program has two sublayers: While the web application layer becomes an interface for video conferencing, the video analyzer layer will process the real time video of the interviewee. The desktop plugins will be used for extraction of necessary data of interviewee and will communicate with the database to send these data.

Database is the layer of storing analysis results, not as a photo but processed and quantified representations of those photos.

Web Server is the layer that becomes a bridge between other layers and manages the data sent and taken from client side and database.

Interviewer client is like the end result of Interly.ai where interviewers can see and control the data of interviewees coming from the video analyzer layer through database and web server layers.

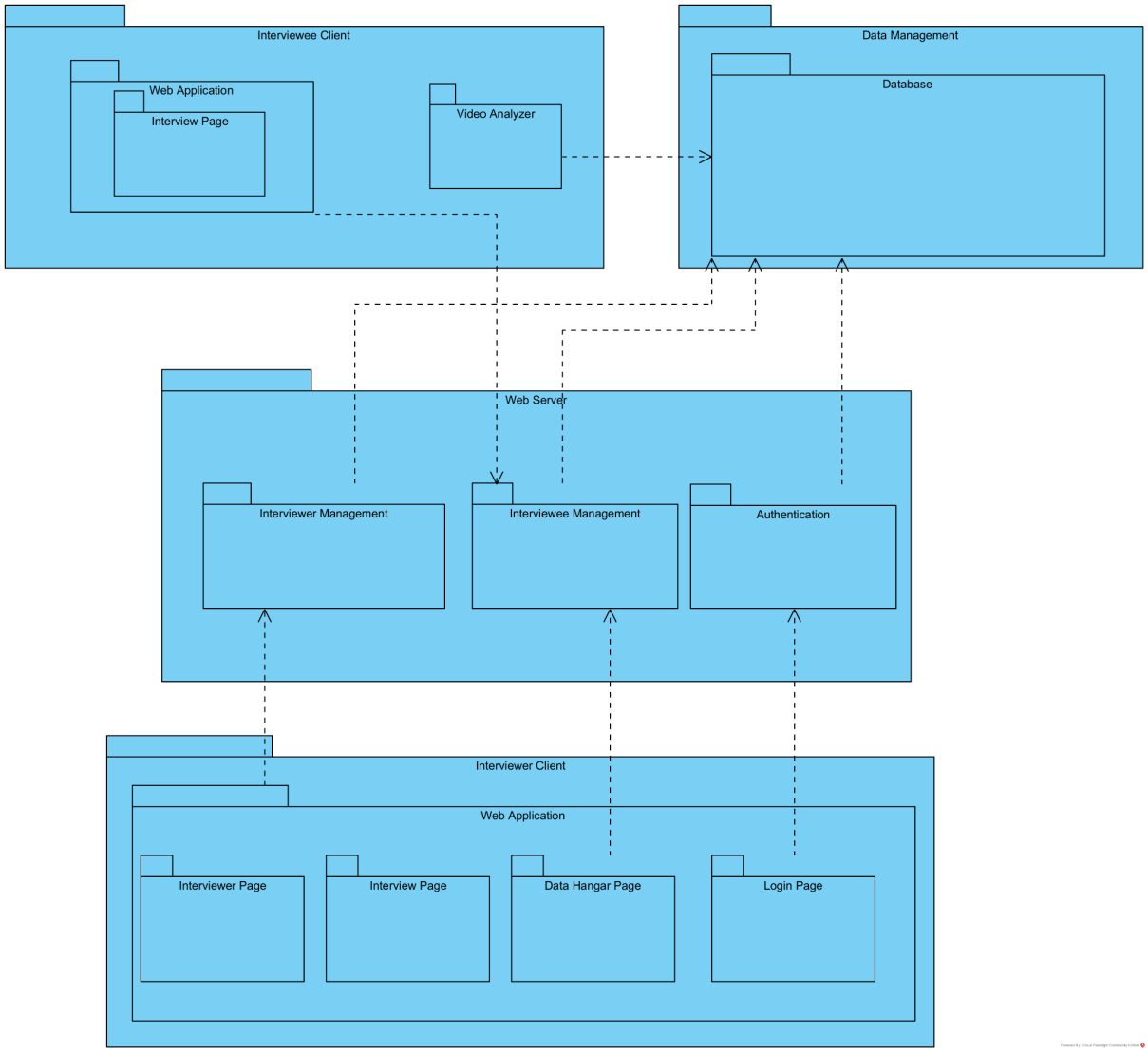

Figure 1: Subsystem Decomposition Diagram

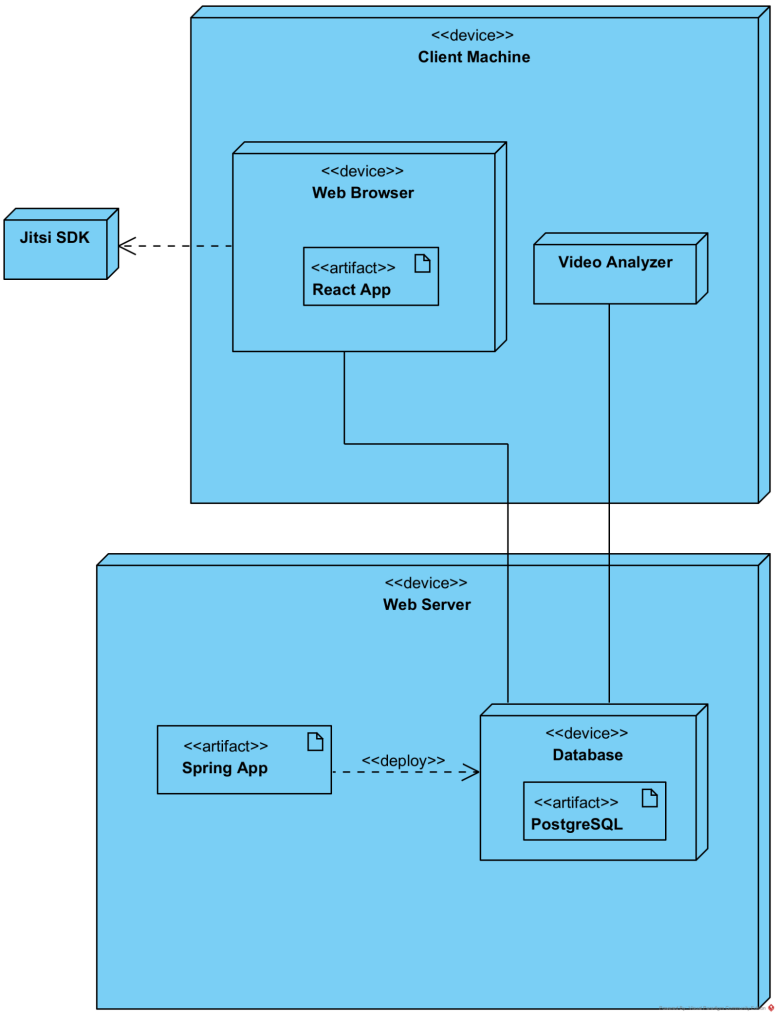

Figure 2: Deployment Diagram

3.3 Hardware/software mapping

Interly.ai is a web application used for online meetings, mostly for interview purposes. The application needs a desktop camera, microphone for communication purposes. More deeply, computer memory, CPU, OS, etc. are used for application running. In the background of application run, Java and Python automatically handles operational services for memory, CPU, IO, and so forth jointly with the operating system.

To keep and store the user data, remote servers will be utilized like AWS servers.

3.4 Persistent data management

Interly.ai has a lot of data for personality analysis. The images of the interviewee are taken every 5 seconds. Even though the images are not stored, the analysis result for every image is stored in the database which causes a huge amount of memory usage. As the duration of the interview increases, the data created in the analysis increases as well. Since the analysis and comparisons will be done by interviewers, and some reviews will be done by HR managers, the data must be persistently stored. Moreover, users', companies', and any other info used in the application should be persistent. Hence, according to the plan, the data is going to be stored in AWS servers and the data management will be held there.

3.5 Access control and security

Since the interviewees' personalities are analyzed in this application, it is extremely essential to provide privacy and security. In order not to face any security gap while analyzing interviewee by his/her images obtained during the interview, the analysis is planned to be done in the interviewee side so that there is no need for sending images to any other side and only the encrypted analysis data is sent to the servers. By doing this, the possibility of any security leak caused by stolen user images is prevented.

On the other hand, there are different types of users in this application such as admins, interviewees, interviewers, companies, etc. Some of them have limited access to different sides of the application. For example, admins have control over the whole system. An interviewer can have access to the results of only his/her interviews but not the others'. An interviewee can only see his/her result. In such a system where users have different access according to their roles, the authentication for the features of the application is limited based on user role. This control is provided by checking the user role during logging in to Interly.ai. Based on the user logged in, the application directs the user with restricted access accordingly in the background.

4. Subsystem services

4.1 Interviewee Client

The Interviewee Client has 2 different subsystems.

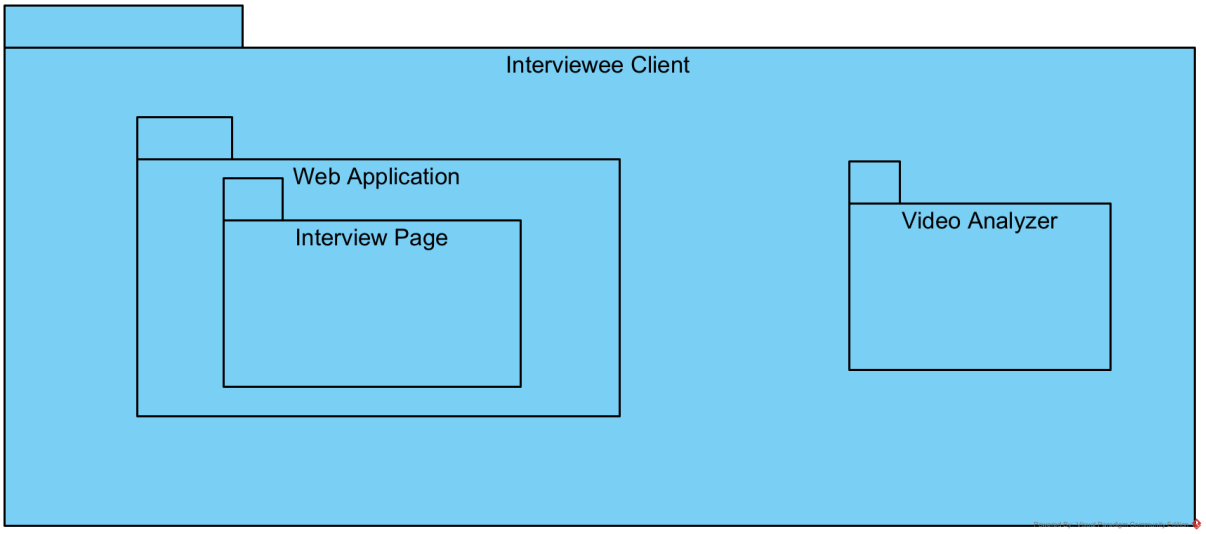

Figure 3: Interviewee Client Subsystem

4.1.1 Web Application

This subsystem is React Application for Interly.ai. This subsystem communicates with the Web Server subsystem through Interviewee Management.

4.1.1.1 Interview Page

This page provides an interface for online meeting calls using Jitsi SDK. This page is a simple online meeting interface for the interviewee client.

4.1.2 Video Analyzer

This subsystem is a deep learning application of Interly.ai. Video Analyzer analyzes Interviewee client in real time and communicates with Data Management Layer through Database subsystem.

4.2 Web Server

This Layer has 3 different management subsystems.

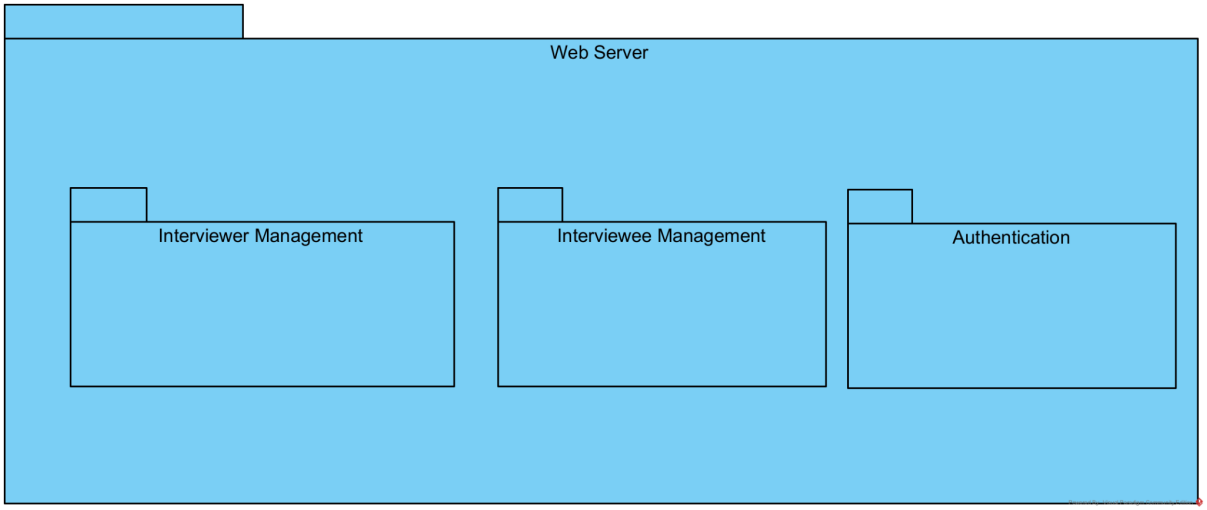

Figure 4: Web Server Subsystem

4.2.1 Interviewer Management

In this layer data of Interviewers are managed by communicating with the Data Management Layer through Database.

4.2.2 Interviewee Management

In this layer data of Interviewees are managed by communicating with the Data Management Layer through Database.

4.2.3 Authentication

This layer manages authentication of Interviewers by checking their login data.

4.3 Interviewer Client

This layer has 4 different interface subsystems

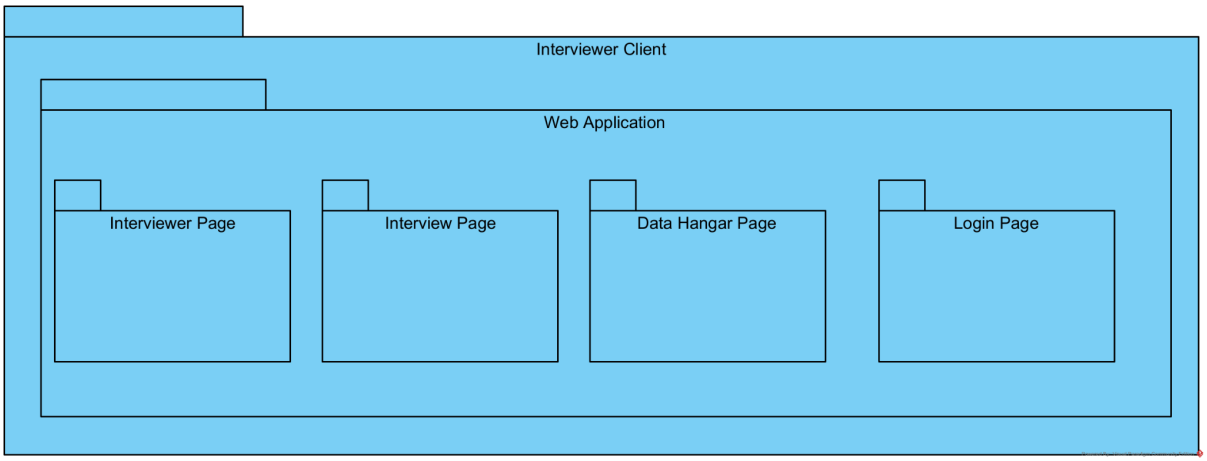

Figure 5: Interviewer Client Subsystem

4.3.1 Interviewer Page

This page provides an interface for details of interviewees. Communicates with a web server to control the information in the database.

4.3.2 Interview Page

This page provides an interface for online meeting calls using Jitsi SDK. This page demonstrates the real time emotion analysis results for the interviewee to the interviewer via interacting with the web server.

4.3.3 Data Hangar Page

This page provides an interface for the meeting results and overall analysis of the interviewee‘s characteristics. Also communicates with the web server to convey these results to the database.

4.3.4 Login Page

Provides the interface for login functionality of interly.ai. Interacts with web server for authentication and tokenization purposes.

4.4 Data Management

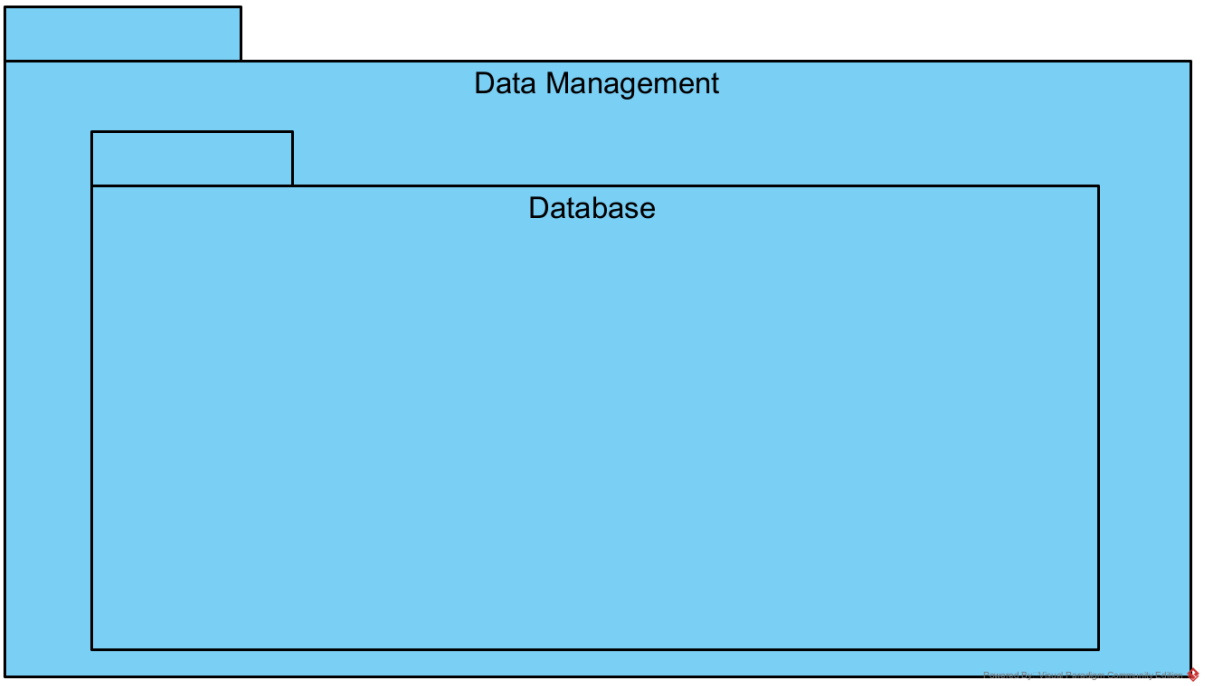

Figure 6: Data Management Subsystem

4.4.1 Database

Database layer is used for storing login info and company account data. Also minimal data about interviewees might be stored to be able to serve proposed functionality.

5. Test Cases

5.1 Functional Test Cases

Test Case ID	TC_InAI_01
Test Case Category	Integration Test
Test Case Objective	Successful Company Login
Precondition	● A valid company account
Procedure	1. In the Sign in screen enter a valid company email. 2. Enter a valid password for the specified valid email. 3. Click the “Login” button.
Test Data	1. A valid company email 2. A valid company password
Expected Result	The user logged in successfully and redirected to the company main menu page.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_02
Test Case Category	Integration Test
Test Case Objective	Successful Interviewer Login
Precondition	● A valid interviewer account
Procedure	1. In the Sign in screen enter a valid interviewer email. 2. Enter a valid password for the specified valid email. 3. Click the “Login” button.
Test Data	1. A valid interviewer email 2. A valid interviewer password
Expected Result	The user logged in successfully and redirected to the interviewer main menu page.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_03
Test Case Category	Integration Test
Test Case Objective	Successful HR Manager Login
Precondition	● A valid HR manager account
Procedure	1. In the Sign in screen enter a valid HR Manager email. 2. Enter a valid password for the specified valid email. 3. Click the “Login” button.
Test Data	1. A valid HR Manager email 2. A valid HR Manager password
Expected Result	The user logged in successfully and redirected to the HR Manager main menu page.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_04
Test Case Category	Integration Test
Test Case Objective	Successful Admin Login
Precondition	● A valid admin account
Procedure	1. In the Sign in screen enter a valid admin email. 2. Enter a valid password for the specified valid email. 3. Click the “Login” button.
Test Data	1. A valid admin email 2. A valid admin password
Expected Result	The user logged in successfully and redirected to the admin main menu page.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_05
Test Case Category	Integration Test
Test Case Objective	Succesful Company Application
Precondition	● A company email which is not registered to the system.
Procedure	1. Click the “Join Us Button” in the Landing page. 2. Fill the form 3. Click Apply button
Test Data	● Company Name ● Company Email ● Name of Applicant ● Number of Employees ● Country ● State ● Post Code ● Any Other Information
Expected Result	The application should be saved to the database successfully
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_06
Test Case Category	Unit Test
Test Case Objective	Not being able to pick overlapping interview start and end time
Precondition	● A valid interviewer account
Procedure	1. Login to the interviewer account 2. From the navbar, click the “Create Interview” button. 3. Choose a start time 4. Try to choose the same time as end time
Test Data	● An overlapping start and end time for interview
Expected Result	The start time can not be picked as end time
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_07
Test Case Category	Unit Test
Test Case Objective	Not being able to pick past days as interview day
Precondition	● A valid interviewer account
Procedure	1. Login to the interviewer account 2. From the navbar, click the “Create Interview” button. 3. Try to pick past days as interview day
Test Data	● A past date for interview
Expected Result	The interview date cannot be a past day (failure in interview creation)
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_08
Test Case Category	Integration Test
Test Case Objective	Not being able to create overlapping interviews for an interviewer
Precondition	● A valid interviewer account
Procedure	1. Login to the interviewer account 2. From the navbar, click the “Create Interview” button. 3. Create an interview with the specified date and time 4. From the navbar, click the “Create Interview” button. 5. Try to create an interview with the same date and time
Test Data	● Interviewee email ● Interviewee name ● Interviewee surname ● Interviewee CV ● Position name ● Interview date ● Start and end time for interview
Expected Result	An interviewee cannot have overlapping interviews (failure in interview creation)
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_09
Test Case Category	Integration Test
Test Case Objective	Joining an interview with a maximum of 15 minutes delay
Precondition	● A valid interviewer account ● An interview
Procedure	1. Login to the interviewer account 2. From the main menu try to join an interview that started 15 minutes ago.
Test Data	● A valid interview
Expected Result	Success in joining the interview
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_10
Test Case Category	Integration Test
Test Case Objective	Joining an interview maximum 15 minutes early
Precondition	● A valid interviewer or HR Manager account ● An interview
Procedure	1. Login to the interviewer or HR Manager account 2. From the main menu try to join an interview that will start 15 minutes later.
Test Data	● A valid interview
Expected Result	Success in joining the interview
Priority	Minor
Date Tested and Result	

Test Case ID	TC_InAI_11
Test Case Category	Unit Test
Test Case Objective	Create a company successfully
Precondition	• A valid company account
Procedure	1. In the “CompanyService.java” file, call create function with the company account.
Test Data	• A valid company email
Expected Result	Save the created company account to the database
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_12
Test Case Category	Unit Test
Test Case Objective	Not being able to create a company with an already existing company email in the system
Precondition	• A valid company account
Procedure	1. In the “CompanyService.java” file, call create function with the company account.
Test Data	• A valid company email
Expected Result	Run time exception with “Company already exists” message.
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_13
Test Case Category	Unit Test
Test Case Objective	Not being able to create a company with an already existing interviewer email in the system
Precondition	● A valid interviewer account
Procedure	1. In the “CompanyService.java” file, call create function with the company account.
Test Data	● A valid interviewer email
Expected Result	Run time exception with “Company already exists” message.
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_14
Test Case Category	Unit Test
Test Case Objective	Not being able to approve a nonexistent company
Precondition	
Procedure	1. In the “CompanyService.java” file, call the approve function with the specified company name.
Test Data	● A nonexistent company email
Expected Result	The function must return false.
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_15
Test Case Category	Unit Test
Test Case Objective	Approve a company successfully
Precondition	● A valid company account
Procedure	1. In the “CompanyService.java” file, call the approve function with the specified company name.
Test Data	● A valid company email
Expected Result	The function must set the “isApproved” variable of the company to true and send a password to the company with the company email.
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_16
Test Case Category	Unit Test
Test Case Objective	Do not approve a company if it is already approved
Precondition	● A valid approved company account
Procedure	1. In the “CompanyService.java” file, call the approve function with the specified company name.
Test Data	● A valid company email
Expected Result	The function must return false.
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_17
Test Case Category	Unit Test
Test Case Objective	Successfully creating default password with at least 2 lower case, 2 upper case, 2 digit, 2 special character and 10 character long
Precondition	
Procedure	1. In the “CompanyService.java” file, call the generatePassayPassword function
Test Data	
Expected Result	A password with given rules
Priority	Minor
Date Tested and Result	

Test Case ID	TC_InAI_18
Test Case Category	Unit Test
Test Case Objective	Successfully joining to a meeting as an interviewee
Precondition	● A valid interview ● A valid interviewee password
Procedure	1. In the “IntervieweeService.java” file, call the “joinMeeting” function
Test Data	● A valid interview id ● A valid interviewee password
Expected Result	The function must return the corresponding interview and set interviewee’s status as true
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_19
Test Case Category	Unit Test
Test Case Objective	Fail in joining to a meeting as an interviewee
Precondition	● A valid interview ● An invalid interviewee password
Procedure	1. In the “IntervieweeService.java” file, call the “joinMeeting” function
Test Data	● A valid interview id ● An invalid interviewee password
Expected Result	The function must return null
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_20
Test Case Category	Unit Test
Test Case Objective	Fail in joining to a meeting as an interviewee
Precondition	● An invalid interview id ● An interviewee password
Procedure	1. In the “IntervieweeService.java” file, call the “joinMeeting” function
Test Data	● An invalid interview id ● An interviewee password
Expected Result	The function must return null
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_21
Test Case Category	Unit Test
Test Case Objective	Create an interviewer successfully
Precondition	● A valid company email
Procedure	1. In the “InterviewerService.java” file, call the “create” function
Test Data	● A valid company email ● Valid interviewer information
Expected Result	An interviewer creation with success
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_22
Test Case Category	Unit Test
Test Case Objective	Not being able to create an interviewer with invalid company email
Precondition	● An invalid company email
Procedure	1. In the “InterviewerService.java” file, call the “create” function
Test Data	● Invalid interviewer information
Expected Result	The function must throw an exception
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_23
Test Case Category	Unit Test
Test Case Objective	Not being able to create an interviewer with invalid email
Precondition	● An invalid interviewer email
Procedure	1. In the “InterviewerService.java” file, call the “create” function
Test Data	● Invalid interviewer information
Expected Result	The function must throw an exception
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_24
Test Case Category	Unit Test
Test Case Objective	Return the stats of all interviewers of a company successfully
Precondition	● A valid company account
Procedure	1. In the “InterviewerService.java” file, call the “getStatsOfAllInterviewers” function with specified company id
Test Data	● A valid company id
Expected Result	The function must return a list containing all of the interviewers and their stats
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_25
Test Case Category	Unit Test
Test Case Objective	Return an exception when the company does not exist in the system
Precondition	
Procedure	1. In the “InterviewerService.java” file, call the “getStatsOfAllInterviewers” function with an invalid company id
Test Data	● An invalid company id
Expected Result	The function must return an exception
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_26
Test Case Category	Unit Test
Test Case Objective	Successfully change an interviewers status from interviewer to HR Manager
Precondition	● A valid interviewer account
Procedure	1. In the “InterviewerService.java” file, call the “changeUserType” function with the specified interviewer id and true as user type
Test Data	● A valid interviewer id ● True as user type
Expected Result	The function must return true and change interviewers status to HR Manager
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_27
Test Case Category	Unit Test
Test Case Objective	Successfully change an interviewers status from HR Manager to interviewer
Precondition	● A valid interviewer account
Procedure	1. In the “InterviewerService.java” file, call the “changeUserType” function with the specified interviewer id and false as user type
Test Data	● A valid interviewer id ● False as user type
Expected Result	The function must return true and change interviewers status to interviewer
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_28
Test Case Category	Unit Test
Test Case Objective	Return an exception when interviewer id does not exist
Precondition	● An invalid interviewer account
Procedure	1. In the “InterviewerService.java” file, call the “changeUserType” function with the specified interviewer id and true or false as user type
Test Data	● An invalid interviewer id ● False or True as user type
Expected Result	The function must return an exception
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_29
Test Case Category	Unit Test
Test Case Objective	Return all interviewers of a company successfully
Precondition	• A valid company account
Procedure	1. In the “InterviewerService.java” file, call the “getAllByCompanyIdAndUserType” function with the specified company id and False as user type
Test Data	• A valid company id • False as user type
Expected Result	Return all of the interviewers of the specified company
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_30
Test Case Category	Unit Test
Test Case Objective	Return all HR Managers of a company successfully
Precondition	• A valid company account
Procedure	1. In the “InterviewerService.java” file, call the “getAllByCompanyIdAndUserType” function with the specified company id and True as user type
Test Data	• A valid company id • True as user type
Expected Result	Return all of the HR Managers of the specified company
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_31
Test Case Category	Unit Test
Test Case Objective	Return an exception when the company does not exist
Precondition	• An invalid company account
Procedure	1. In the “InterviewerService.java” file, call the “getAllByCompanyIdAndUserType” function with the specified company id and True or False as user type
Test Data	• An invalid company id • True or False as user type
Expected Result	The function must return an exception
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_32
Test Case Category	Unit Test
Test Case Objective	Return all interviews of a department in the company successfully
Precondition	• A valid company account • A valid department name
Procedure	1. In the “InterviewerService.java” file, call the “getInterviewsByCompanyIdAndDepartment” function with the specified company id and department name
Test Data	• A valid company id • A valid department name
Expected Result	The function must return a list of all interviews of the specified department in the company
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_33
Test Case Category	Unit Test
Test Case Objective	Return an exception when the department name is invalid
Precondition	• A valid company account • An invalid department name
Procedure	1. In the “InterviewerService.java” file, call the “getInterviewsByCompanyIdAndDepartment” function with the specified company id and department name
Test Data	• A valid company id • An invalid department name
Expected Result	The function must return an exception
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_34
Test Case Category	Unit Test
Test Case Objective	Return an exception when the company is invalid
Precondition	• An invalid company account • An valid department name
Procedure	1. In the “InterviewerService.java” file, call the “getInterviewsByCompanyIdAndDepartment” function with the specified company id and department name
Test Data	• A valid company id • An invalid department name
Expected Result	The function must return an exception
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_35
Test Case Category	Unit Test
Test Case Objective	Return all interviews of an interviewer successfully
Precondition	• A valid interviewer account
Procedure	1. In the “InterviewService.java” file, call the “getInterviewsByInterviewerId” function with the specified interviewer id
Test Data	• A valid interviewer id
Expected Result	Return a set of all interviews of a specified interviewer
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_36
Test Case Category	Unit Test
Test Case Objective	Throw an exception when the interviewer id is not valid
Precondition	• An invalid interviewer account
Procedure	1. In the “InterviewService.java” file, call the “getInterviewsByInterviewerId” function with the specified interviewer id
Test Data	• A valid interviewer id
Expected Result	The function must return an exception
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_37
Test Case Category	Unit Test
Test Case Objective	Create a payment for a company successfully
Precondition	• Payment information
Procedure	1. Call “create” function in the “paymentService.java” file
Test Data	• Payment Information
Expected Result	The function must save the payment and set the payment as companies payment
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_38
Test Case Category	Unit Test
Test Case Objective	Return an exception when company id is invalid
Precondition	• Payment information
Procedure	1. Call “create” function in the “paymentService.java” file
Test Data	• Payment Information
Expected Result	The function must return an exception
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_39
Test Case Category	Unit Test
Test Case Objective	Return companies all payments successfully
Precondition	● A valid company account
Procedure	1. Call “create” function in the “getAllPaymentsByCompanyId.java” file with the company id
Test Data	● A valid company id
Expected Result	Return all payments of a company
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_40
Test Case Category	Unit Test
Test Case Objective	Throw an exception when company account is not valid
Precondition	● An invalid company account
Procedure	1. Call “create” function in the “getAllPaymentsByCompanyId.java” file with the company id
Test Data	● An invalid company id
Expected Result	Return an exception
Priority	Major
Date Tested and Result	

5.2 Non-Functional Test Cases

Test Case ID	TC_InAI_41
Test Case Category	Non-Functional (Security)
Test Case Objective	Application redirects the user to login screen when URL copied and pasted to a different window
Precondition	
Procedure	1. Copy the URL when logged in and past it to a different browser or window
Test Data	● URL
Expected Result	When logged in, copying and pasting the URL to a different browser or window should redirect the user to the login screen.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_42
Test Case Category	Non-Functional (Security)
Test Case Objective	Users should not be able to copy and paste the passwords.
Precondition	● A valid account
Procedure	1. Copy and paste the password into the password input box
Test Data	● Password
Expected Result	The copy paste should not be performed in the password input box.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_43
Test Case Category	Non-Functional
Test Case Objective	Users should not be able to see their session information in the browser console.
Precondition	● A valid account
Procedure	1. Login to a valid account 2. Open inspect 3. Open application window 4. Open local storage section
Test Data	1. A valid email 1. A valid password
Expected Result	The sensitive session information should not be readable or seen.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_44
Test Case Category	Non-Functional (Security)
Test Case Objective	Users should not be able continue using the application when they logout.
Precondition	● A valid account
Procedure	1. Login to a valid account 2. Logout from an account 3. Click the back button
Test Data	1. A valid email 1. A valid password
Expected Result	The application must redirect the user to the login screen.
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_45
Test Case Category	Non-Functional (Usability)
Test Case Objective	The interviewee should be able to use the application without any lag and performance losses
Precondition	● A valid interviewee ● A valid interview
Procedure	1. Create an interview 2. Join the interview as interviewee 3. Join the interview as interviewer 4. Check errors and freezes
Test Data	1. A valid email 1. A valid password 2-3. A valid interview password
Expected Result	There should not be any connection loss because of the application and freezes in the video conference and calculations
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_46
Test Case Category	Non-Functional (Performance-Load)
Test Case Objective	The users should be able to the application under 50 user load
Precondition	
Procedure	1. Load the application with 50 user load where each user represents 100 request per second
Test Data	
Expected Result	There should be little to no severe errors and lags in the application response
Priority	Critical
Date Tested and Result	

Test Case ID	TC_InAI_47
Test Case Category	Non-Functional (Performance-Load)
Test Case Objective	The users should be able to the application under 100 user load
Precondition	
Procedure	1. Load the application with 100 user load where each user represents 100 request per second
Test Data	
Expected Result	There should be little to no severe errors and lags in the application response
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_48
Test Case Category	Non-Functional (Performance-Stress)
Test Case Objective	Determine the breaking point of the application
Precondition	
Procedure	1. Start with 50 user load where each user represents 100 request per second and increase the load regularly until the application could not response to the requests
Test Data	
Expected Result	The breaking point should be at least 1000 user load
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_49
Test Case Category	Non-Functional (Performance-Volume)
Test Case Objective	Measure the loss of performance respect to data volume
Precondition	
Procedure	1. Determine when the errors and unresolved requests start to appear while increasing the data volume on database
Test Data	
Expected Result	The application should be able to perform well until the data volume becomes equal to 5 years of data volume
Priority	Major
Date Tested and Result	

Test Case ID	TC_InAI_50
Test Case Category	Non-Functional (Recovery)
Test Case Objective	Determine when the application becomes available after a failure
Precondition	
Procedure	1. Measure the time spent while waiting for the application to become available again
Test Data	
Expected Result	At most 2 hours should be needed to recover the application
Priority	Major
Date Tested and Result	

6. Consideration of Various Factors in Engineering Design

The right to erasure or right to be forgotten is one of the important constraints on the project [5]. According to this right, an interviewee may want to delete his/her CV and analyses saved in the database. Therefore, an interviewee should be allowed to ask to be deleted. In order to achieve this, admins have the right to delete any

information of an interviewee. If an interviewee wants to be deleted, the interviewee must send an email to the companies they have been applied to or to the admins to delete all information. If they send the email to a company, then the company must send an email to the admins in order to ask them to delete the interviewee from the company's database. After that, admins can delete the information of the interviewee.

In a company, vertical hierarchy is important. Since it increases efficiency by allowing people who are higher in the hierarchy to check whether their subordinates are working efficiently and correctly. Therefore, HR managers should see a report about each human resources staff. The report includes recruitment rate and other statistics. With these reports, HR managers can compare each staff and determine their efficiency.

Factors	Level Of Effect	Effects
Privacy	4	Admins can delete any information of an interviewee.
Economy	6	HR managers can view statistics of each interviewer and compare them.

7. Teamwork Details

7.1 Contributing and functioning effectively on the team

Equal distribution of tasks is an important criterion for this project. Therefore, each member of the team takes responsibility when necessary. As a team, we split into two different domains for some time as an user interface and a deep learning domain. In this way, it was aimed to advance the project from two different branches quickly. At the end of this process, the teams shared their progress with each other and the people who would take part in the teams were determined again. In this way, the member in one domain will not continue the project in a very disconnected way from the other domain. In addition, if there is an area in which each member in the team specializes, it is planned to take quick action when a job needs to be done in

their own field. In this way, each member in the team shares the leadership. These sprints will continue for the rest of the semester for different domains such as backend or server domains. Furthermore, feedback is an important point in the development of this project since it reinforces the positive communication and it enhances the performance of the team. Through feedback, each member feels the strength of being in a team. A team member takes the responsibility of another member for some time if this member is not available to handle his/her task, which is the positive outcome of the proper teamwork. In the rest of the semester, the following tasks with their corresponding assignees are as follows:

Task No	Title	Leader	Members	Due Date
1	Deep Learning: Emotion Recognition Implementation Enhancements	Murat Furkan Uğurlu	Furkan Turunç, Osman Semih Tiryaki, Erhan Er, Arda Serim	April 10
2	Deep Learning: Personality Test Implementation	Furkan Turunç	Murat Furkan Uğurlu, Osman Semih Tiryaki, Erhan Er, Arda Serim	April 10
3	Deep Learning: Stress Detection	Osman Semih Tiryaki	Murat Furkan Uğurlu, Furkan Turunç, Erhan Er, Arda Serim	April 20
4	Deep Learning: Self-confidence Detection	Erhan Er	Murat Furkan Uğurlu, Osman Semih Tiryaki, Furkan Turunç, Arda Serim	April 20
5	Performance Tuning: Improvements on Real-Time Graphs in Interview	Arda Serim	Murat Furkan Uğurlu, Erhan Er	April 20
6	Frontend: Changes and Arrangements	Erhan Er	Arda Serim, Murat Furkan Uğurlu	May 1
7	Backend: Changes and Arrangements	Arda Serim	Osman Semih Tiryaki, Furkan Turunç	May 1
8	Prepare Executable	Osman Semih Tiryaki	Furkan Turunç, Murat Furkan Uğurlu	May 10

7.2 Helping creating a collaborative and inclusive environment

Each member in the team collaborated to their domain and the other domain as well. Whenever help is needed from one side, the other domain team tries to give support in order to remove the blockage. Furthermore, interior domain cooperation is handled equally and the teams do not try to overload a person more than others.

7.3 Taking lead role and sharing leadership on the team

The team generally shared equal responsibility for the project and the leadership was not taken by any member nominately. Even though it might cause some minor bottlenecks in the course of the project, it forces each team member to say a word to shape the project and eliminates the possibility of the realization of only one person’s wishes within the project. Therefore, each team member effectively takes responsibility and improves himself on the management cycle of a project.

8. Glossary

UI: User Interface
Data Hangar: All data of analysis of an interviewee in his/her interview.

9. References

[1] “How covid-19 has fundamentally changed recruitment,” *EURES*, 13-Oct-2021. [Online]. Available: https://eures.ec.europa.eu/how-covid-19-has-fundamentally-changed-recruitment-2021-10-13_en. [Accessed: 13-Nov-2022].

[2] V. Postrel, “Job interviews should stay online after covid,” *Bloomberg.com*, 07-Apr-2021. [Online]. Available: https://www.bloomberg.com/opinion/articles/2021-04-07/job-interviews-should-stay-online-after-covid. [Accessed: 13-Nov-2022].

[3] “Our Solution”, *VireUp*. [Online]. Available: https://www.vireup.com/our_solution/. [Accessed: 10-Mar-2022].

[4] “Intelligent Talent Acquisition System”, *easyhire*. [Online]. Available: https://easyhire.me/ [Accessed: 12-Nov-2022].

[5] “Right to erasure,” *ICO*. [Online]. Available: https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/individual-rights/right-to-erasure/#:~:text=individual%20for%20ID%3F-,What%20is%20the%20right%20to%20erasure%3F,time%20the%20request%20is%20received . [Accessed: 12-Nov-2022].