Bilkent University

Department of Computer Engineering

Senior Design Project

Interly.ai

Final 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

May 19, 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.

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 current mood, level of anxiety, and deception are presented to the interviewer's screen real-timely.

2. Requirement Details

2.1 Functional Requirements

2.1.1 System Requirements

The system should:

- register companies when they send a registration form
- allow companies to fill the registration form
- save companies to database when they are approved
- make a login system for HR managers and Interviewees
- allow HR managers to change the role of interviewers
- allow interviewers to see data hangars of interviewees
- create a video conferencing room
- allow interviewee and interviewers to make an online meeting
- show the interviewee's emotional analysis, anxiety and deceit interpretations
- show the real time and average analysis information graphics

2.1.2 Interviewer Requirements

Interviewers should be able to:

- access the website
- login to interviewer account
- review upcoming meetings
- review potential candidates information
- start a meeting
- give a name to meeting room
- send an invitation to interviewee
- send necessary information to interviewee
- make a video conference call with interviewee
- share screen
- enlarge participant display panel
- turn on/off microphone
- turn on/off camera
- remove participants from meeting room
- mute the participants
- see real time emotional analysis of interviewee in real time emotion graph
- see average emotional analysis of interviewee in average emotion graph
- see stress level of interviewee
- see probability of deception
- end the meeting
- choose to preview data hangar feature
- see average emotional and stress level data about the interviewee
- see data hangars for past interviewees
- compare the emotional analysis of interviewees applied for same role

2.1.3 Interviewee Requirements

Interviewees should be able to:

- access to our website
- download the installer for plugins
- run the installer
- install the plugins
- run the executable

- enter the meeting with the information coming from interviewer
- see the personal emotional data analysis terms
- verify that he/she accepts the terms of use
- turn on/off microphone
- turn on/off camera
- leave meeting

2.1.4 HR Manager Requirements

HR Managers should be able to:

- have access to all capabilities of interviewees
- access data hangars of all interviews
- inspect specific interviewer's past interviews
- promote a regular interviewee to HR manager
- demote an HR manager to interviewee

2.1.5 Admin Requirements

Admin should be able to:

- have access to all capabilities that HR managers, interviewees and interviewers capabilities
- add company accounts
- add hr manager accounts
- add interviewee accounts
- remove company accounts
- remove hr manager accounts
- remove interviewee accounts
- approve company registration request
- decline company registration request
- activate company account after first payment
- deactivate company account

2.2 Non-Functional Requirements

2.2.1 Reliability

The system should:

- give analysis data of the interviewee with the highest possible accuracy.
- create a stable video conferencing environment that flow ow meeting is not disrupted by technical problems

2.2.2 Privacy

The system should:

- process sensitive data in users machine and send only numerical results to server
- Store as little sensitive data as possible in database
- Inform the user about which data about him is used and how it is processed
- Inform the user about which data about him is stored in database
- not leak any sensitive data

2.2.3 Efficiency

The system should:

- show the real time emotional, anxiety and deceit analysis of the interviewee.
- not have any delays or crushes

2.2.4 Usability

The system should:

- be user friendly
- not have a complicated design
- have necessary explanation of every feature

2.2.5 Scalability

The system should:

- have enough server capability to serve multiple interviews
- have enough database storage space to serve multiple companies

3. Final Architecture and Design Details

3.1 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.

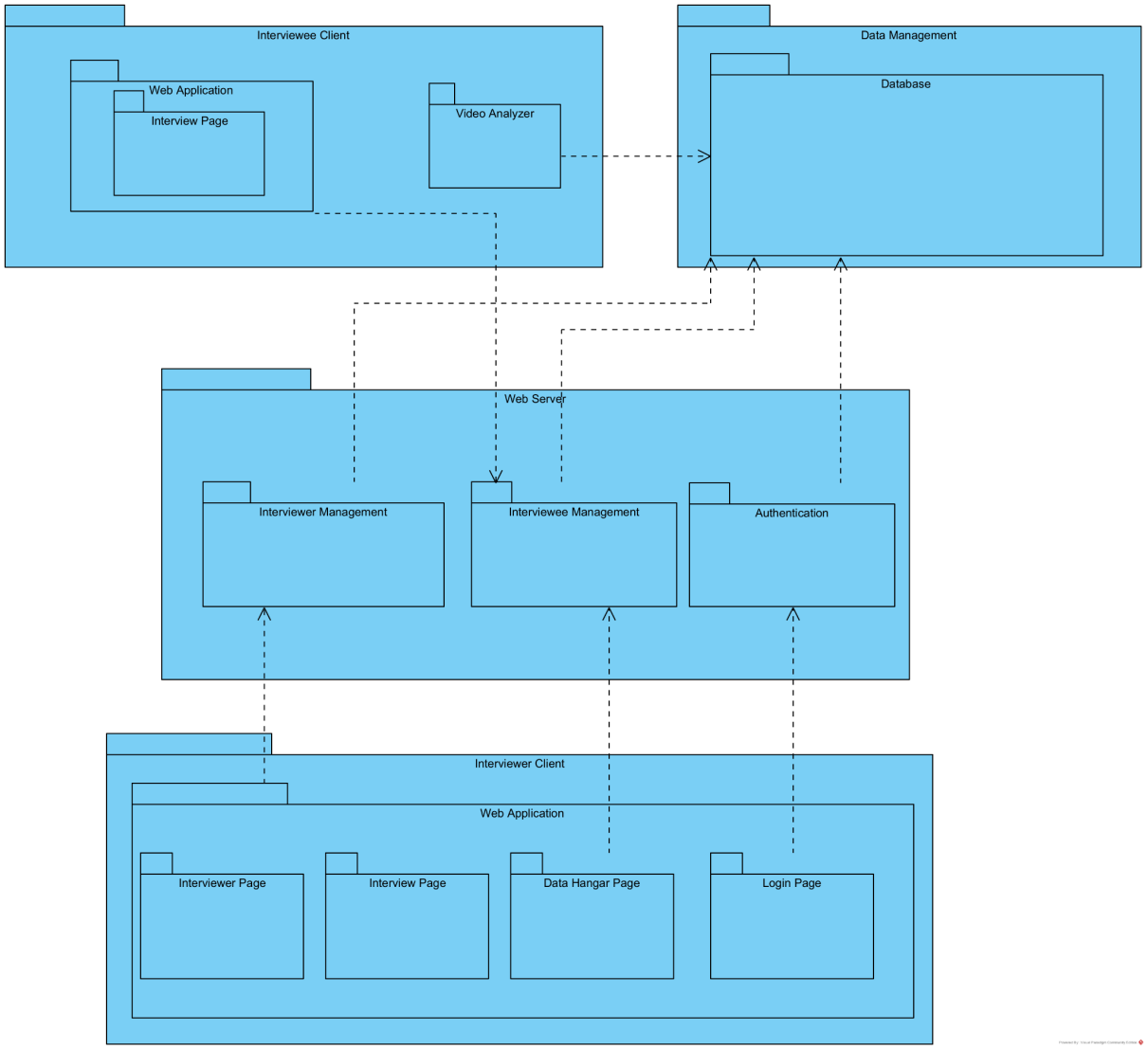


Figure 1: Subsystem Decomposition Diagram

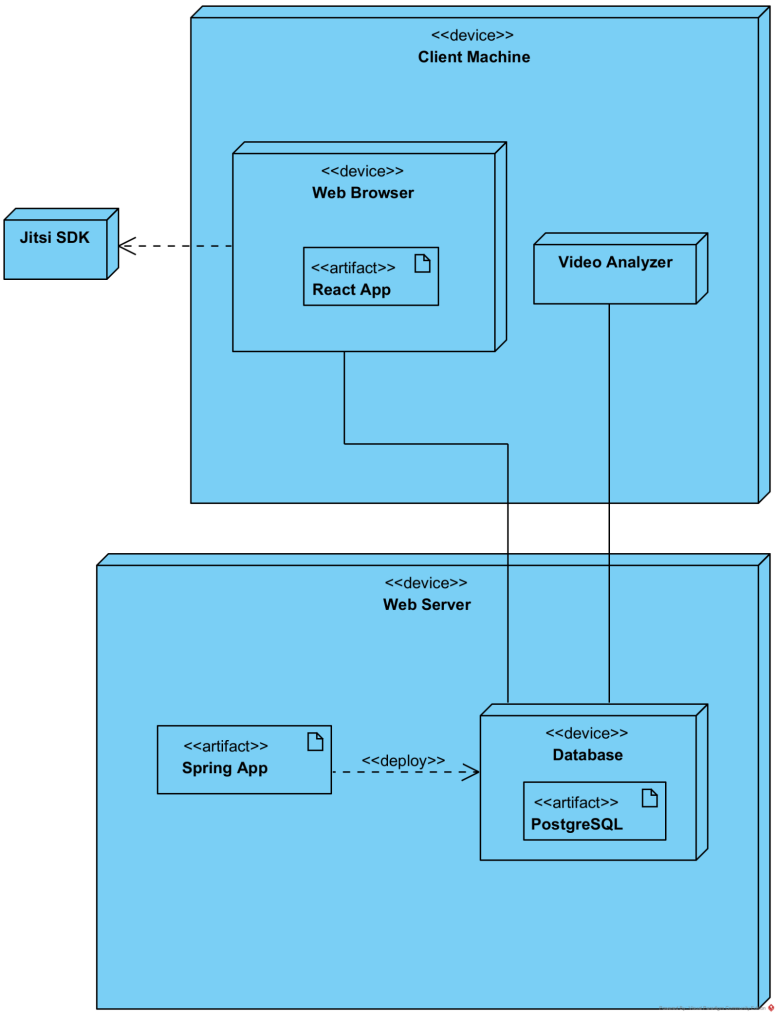


Figure 2: Deployment Diagram

3.1.1 Interviewee Client

The Interviewee Client has 2 different subsystems.

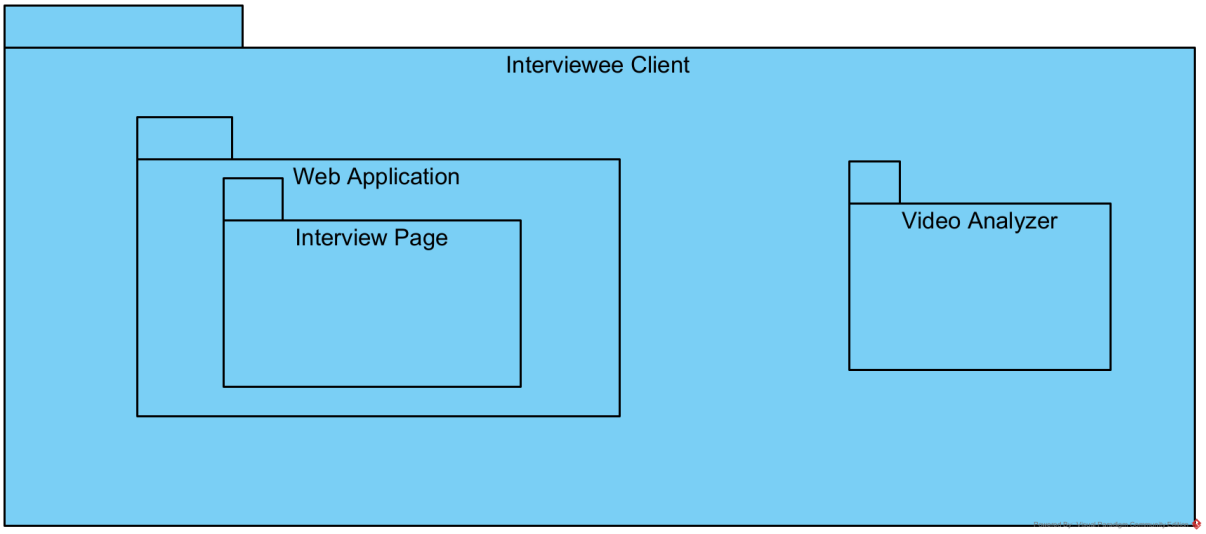


Figure 3: Interviewee Client Subsystem

3.1.1.1 Web Application

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

Interview Page

This page provides an interface for online meeting calls using Jitsi SDK [3]. This page is a simple online meeting interface for the interviewee client, which has basic features like sharing screen, muting microphone, turning off camera, etc.

3.1.1.2 Video Analyzer

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

3.1.2 Web Server

This Layer has 3 different management subsystems.

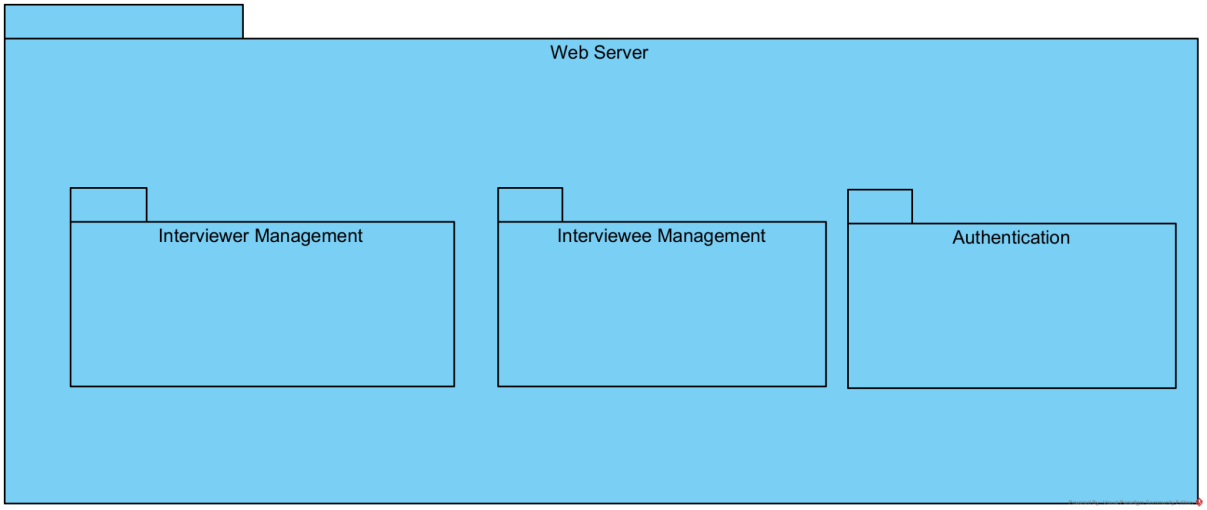


Figure 4: Web Server Subsystem

3.1.2.1 Interviewer Management

In this layer data of Interviewers, like position of the interviewer and interviews they created are managed by communicating with the Data Management Layer through Database.

3.1.2.2 Interviewee Management

In this layer data of Interviewees, such as their analysis results and email addresses are managed by communicating with the Data Management Layer through Database.

3.1.2.3 Authentication

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

3.1.3 Interviewer Client

This layer has 4 different interface subsystems

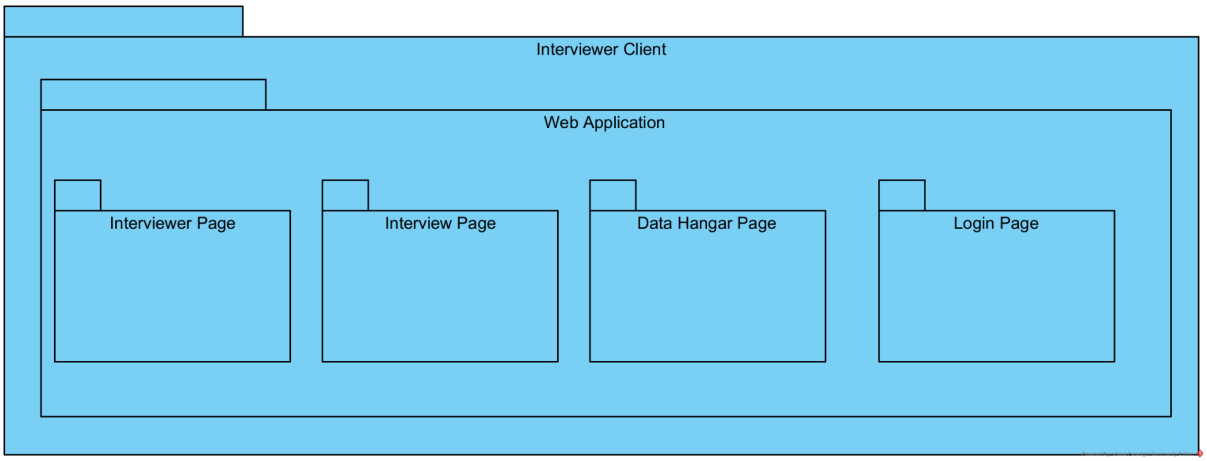


Figure 5: Interviewer Client Subsystem

3.1.3.1 Interviewer Page

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

3.1.3.2 Interview Page

This page provides an interface for online meeting calls using Jitsi SDK [3]. Besides, this page demonstrates the real time emotion, anxiety and deceit analysis results for the interviewee to the interviewer via interacting with the Database in the Data Management Layer.

3.1.3.3 Data Hangar Page

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

3.1.3.4 Login Page

Provides the interface for login functionality of interly.ai. Interacts with Data Management Layer for authentication and tokenization purposes.

3.1.4 Data Management

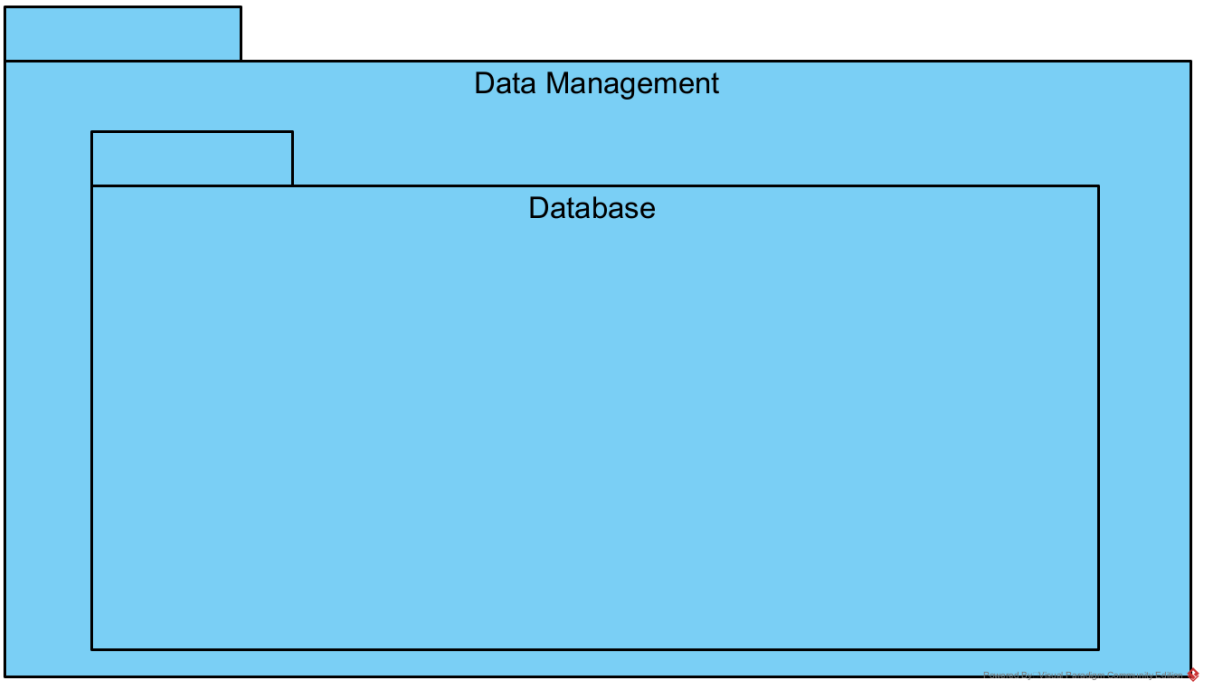


Figure 6: Data Management Subsystem

3.1.4.1 Database

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

3.2 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 Docker servers are utilized.

3.3 Persistent data management

Interly.ai has a lot of data for personality analysis. The images of the interviewee are taken every second. 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 a remote Docker server.

3.4 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.

3.5 Boundary Conditions

3.5.1 Initialization

Initialization boundary conditions pertain to the conditions under which the Interly.ai application starts.

- Users must have an active internet connection to be able to access the application.
- Upon opening the application, the interviewer is asked to log in. The login must be authenticated.
- For an interview analysis to start, the camera of the interviewee must be turned on.

3.5.2 Failure

Failure boundary conditions relate to the conditions under which the system stops working or encounters an error.

- If the network connection fails during an interview, the system should attempt a reconnection. If reconnection fails, the system saves the analysis results up to failure point.
- If the database encounters an issue while storing or retrieving data, an appropriate error message should be displayed, and the system should attempt to resolve the issue.

3.5.3 Termination

Termination boundary conditions apply to the conditions under which the system properly shuts down or ends a session.

- After an interview, the session is finished and all analysis data related to the session is securely saved in the database. The virtual room is then closed.

- If the system needs to be shut down for maintenance or other reasons, any active sessions are saved and users are properly notified of the shutdown.

4. Development/Implementation Details

Interly.ai is a web application that has both frontend and backend development. The implementation details of the frontend and backend can be seen in the first and second part below. The app also has feature side development which is mentioned in the third part.

4.1 Frontend

React.js is used for frontend development. There are some pages categorized under different holders.

Admin Pages	**Description**
AdminApproveCompanyApplication.js	With this script, the admin can see the registration application of a company and can approve them in the UI.
AdminDeleteAccountsPage.js	With this script, the admin can delete any account registered in the application in the UI.
AdminMainMenu.js	With this script, the admin can see all the companies and all pending approvals in the UI.

Company Pages	**Description**
AllHRMs.js	With this script, a company can see all HR managers listed in the UI.
AllHRs.js	With this script, a company can see all HRs (interviewers) in the UI.

CompanyMainPage.js	With this script, a company can see all HR Managers and interviewers in its main page in the UI.
InterviewerCreationPage.js	With this script, a company can create an account for the interviewer by entering information about its interviewer in the UI.

HRM Pages	**Description**
AllDataHangars.js	With this script, all data hangars can be seen by the HR manager in a company in the UI.
HRMMainPage.js	With this script, a HR Manager in a company can see all interviews along with their corresponding data hangars.
InterviewerDetailPage.js	With this script, a HR Manager in a company can see all the recruitments of an interviewer in the UI.

Other Scripts	**Description**
CompanyApplicationFormPage.js	With this script, companies can apply for registering to the application in the UI.
DataHangarPage.js	With this script, data of the interview including interviewee information and the result can be seen in the UI.
InterviewCreationPage.js	With this script, an interviewer can create an interview with the specified time slot in the UI.
InterviewDetailPage.js	With this script, interview details can be seen in the UI.

IntervieweePage.js	With this script, an interviewee can join the meeting in the UI.
InterviewerMainPage.js	With this script, an interviewer can see the data hangars of his/her previous interviews in the UI.
JitsiPage.js	With this script, Jitsi video conferencing meetings can be integrated in the UI.
LandingPage.js	With this script, the landing page of the app can be seen in the UI.
SignInPage.js	With this script, any account owner can sign in to the application in the UI.
UpcomingInterviews.js	With this script, upcoming interviews can be seen by the interviewers in the UI.
UserSettings.js	With this script, the users can change their password in the UI.

4.2 Backend

Spring is used for backend service and PostgreSQL is used for the database. The underlying library is Lombok which is used for data persistence operations without any implementation of the functions in the project. We have used the passay library to generate unique and random passwords for the users in the application. We also have used the JavaMailSender library to send mail about meetings and registrations. Hence, calling the functions in Lombok is enough for the execution. We also have used BCryptPasswordEncoder for the encryption of the password, which means in the database, no one can see the actual passwords, instead, the hashed version of the passwords can be seen. We have used Entity, Repository, Service, and Controller Classes in the project. The architecture is as follows:

Entity Class: represents a domain entity and contains fields, constructors, and methods related to the fields. Entity classes can be listed as follows:

class Admin
It represents the Admin entity class.
Attributes
private Long id
private String userType
private String email
private String password
Functions
Getter/Setter functions

class AnalysisData
It represents the AnalysisData entity class.
Attributes
private Long id
private PersonalityData personalityData
private float levelOfAnxiety
private float levelOfDeceit
Functions
Getter/Setter functions

class Company
It represents the Company entity class.
Attributes

private Long id
private String userType
private String companyEmail
private String companyPassword
private boolean isApproved
Functions
Getter/Setter functions

class CurrentMoodData
It represents the Admin entity class.
Attributes
private Long id
private float levelOfFear
private float levelOfSadness
private float levelOfHappiness
private float levelOfSurprise
private float levelOfAnger
private float levelOfNeutral
private float levelOfDisgust
Functions
Getter/Setter functions

class DataHangar

It represents the DataHangar entity class.
Attributes
private Long id
private Timestamp timeCreated
private String jobPosition
private String intervieweeResult
Functions
Getter/Setter functions

class Interview
It represents the Interview entity class.
Attributes
private Long id
private Timestamp scheduledTime
private String meetingLink
private String meetingPassword
private boolean permissionForAnalysis
Functions
Getter/Setter functions

class Interviewee
It represents the Interviewee entity class.
Attributes

private Long id
private String userType
private String intervieweeEmail
private String intervieweeFirstName
private String intervieweeLastName
private String appliedPosition
private String cv
private boolean status
Functions
Getter/Setter functions

class Interviewer
It represents the Interviewer entity class.
Attributes
private Long id
private String userType
private String interviewerEmail
private String interviewerFirstName
private String interviewerLastName
private String interviewerPassword
private String department
Functions
Getter/Setter functions

class Payment
It represents the Admin entity class.
Attributes
private Long id
private float paymentAmount
private Timestamp paymentTime
Functions
Getter/Setter functions

class Subscription
It represents the Admin entity class.
Attributes
private Long id
private String subscriptionType
private Date startingDate
private boolean isActive
Functions
Getter/Setter functions

Repository Interface: communicates with the underlying database. Repository interfaces can be listed as follows:

RepositoryInterface	Functions
AdminRepository	findById(id), findAdminByEmail(email)
AnalysisDataRepository	findById(id), findAll()
CompanyRepository	findById(id), findByCompanyEmail(companyEmail), findByCompanyName(name), deleteCompanyByCompanyEmail(email)
CurrentMoodDataRepository	findById(id), findAll(), findByAnalysisDataId(id)
DataHangarRepository	findById(id), findAll()
HRManagerRepository	findById(id), findAll(), findAllByCompanyId(companyId)
IntervieweeRepository	findById(id), findAll(), findByIntervieweeEmail(email), deleteByIntervieweeEmail(email)
InterviewerRepository	findById(id), findAll(), findAllByCompanyId(id), findAllByCompanyIdAndUserType(companyId, userType), findInterviewerByInterviewerEmail(email), deleteByInterviewerEmail(email), findAllByCompanyIdAndDepartment(id, department)
InterviewRepository	findById(id), findAll(), findAllInterviewsByIntervieweeIdAndScheduledTime(intervieweeId, scheduledTime), findAllInterviewsByHostIdAndScheduledTime(hostId, scheduledTime), findAllByHostId(id)
	findById(id),

PaymentRepository	findAllPaymentsBySubscriptionId(subscriptionId)
SubscriptionRepository	findById(id)

Service Class: implements business logic and orchestrates operations using one or more entities and repositories. Each controller class has its service class with the implemented version of the functions of the controller class like login, create, delete, get, etc.

Controller Class: handles incoming api related requests, interacts with the service class and returns appropriate responses. Each entity class has its controller with the api requests like login, create, delete, get, etc.

4.3 Feature Side Development

The application has 3 features which are emotion recognition, stress detection, and deception detection. The following subtitles are specialized for these 3 features separately:

4.3.1 Emotion Recognition

In the first phase, we have used the FER library for this task [4]. Because we want to analyze a real-time video of the interviewees, we integrate the code behind the FER library into our project by changing the code when needed. In the second phase, we tried to write our own model by training a Facial Emotion Recognition dataset. In this model, in each frame, we detect the mood and mark it as 1, and other types of moods are signed as 0. Emotion types are happy, sad, neutral, disgust, surprised, fear, and angry. Because this model's accuracy is very low for the test set (explained below), we selected the first model for this part. In this built-in model, the dataset named fer2013 is loaded first. Then using the following functions, the emotion scores for each emotion type is obtained.

Function	Description
get_max_faces(data: list) -> int	This function gets max number of faces detected in a series of frames, eg 3
to_pandas(self, data: Union[pd.DataFrame, list]) -> pd.DataFrame	This function converts results to pandas DataFrame
get_emotions(df: pd.DataFrame) -> list	This function gets emotion columns from results.
analyze(self, detector, # fer.FER instance display: bool = False, output: str = "csv", frequency: Optional[int] = None, max_results: int = None, save_fps: Optional[int] = None, video_id: Optional[str] = None, save_frames: bool = False, save_video: bool = False, annotate_frames: bool = True, zip_images: bool = True, detection_box: Optional[dict] = None,) -> list:	Ths function recognizes facial expressions in video using `detector`.

Training our model

As mentioned earlier, even though we are using a pre-trained model that has almost state of the art level accuracy rates, this was not our first choice. We have trained two different models on different datasets [5]. The first model we had trained was using CNN with pytorch library. That model was our first try about the machine learning models and it was not functioning properly. The accuracy results for that one was very low, thus we will not include the results we get from that model numerically. On the other hand the second model we trained achieved much better and acceptable results on the training set. With an accuracy rate of 94%, our model was much better than the first try. This model was using keras and tensorflow libraries and has a different implementation than pytorch.

```
Epoch 44/50
448/448 [==============================] - 18s 40ms/step - loss: 0.2029 - accuracy: 0.9272
Epoch 45/50
448/448 [==============================] - 18s 39ms/step - loss: 0.2039 - accuracy: 0.9285
Epoch 46/50
448/448 [==============================] - 18s 39ms/step - loss: 0.1930 - accuracy: 0.9315
Epoch 47/50
448/448 [==============================] - 18s 39ms/step - loss: 0.1833 - accuracy: 0.9364
Epoch 48/50
448/448 [==============================] - 18s 40ms/step - loss: 0.1762 - accuracy: 0.9379
Epoch 49/50
448/448 [==============================] - 18s 40ms/step - loss: 0.1713 - accuracy: 0.9398
Epoch 50/50
448/448 [==============================] - 18s 41ms/step - loss: 0.1683 - accuracy: 0.9409
```

However, the reason we did not integrate this model into our system was the accuracy rates in validation and test datasets were not as high as we expected. The results showed that we had around 60% validation and test accuracy rates.

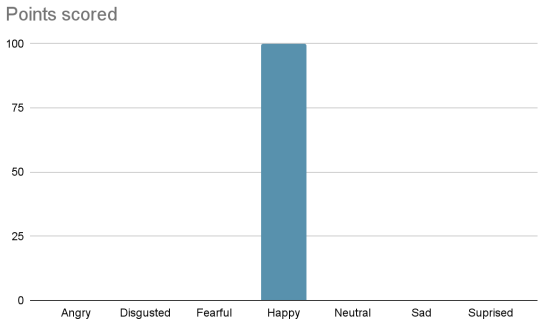
```
val_loss: 1.6057 - val_accuracy: 0.6132

val_loss: 1.5552 - val_accuracy: 0.6194

val_loss: 1.6142 - val_accuracy: 0.6055

val_loss: 1.7041 - val_accuracy: 0.6004

val_loss: 1.6483 - val_accuracy: 0.6167

val_loss: 1.7001 - val_accuracy: 0.6068

val_loss: 1.6944 - val_accuracy: 0.6147

val_loss: 1.7169 - val_accuracy: 0.6158

val_loss: 1.7299 - val_accuracy: 0.6107

val_loss: 1.8043 - val_accuracy: 0.6040

val_loss: 1.8406 - val_accuracy: 0.6092

val_loss: 1.9310 - val_accuracy: 0.5950
```

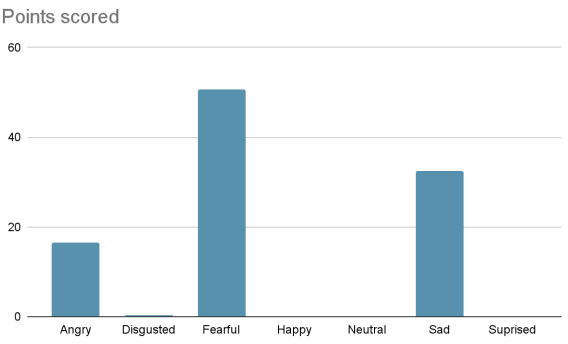
Test accuracy:

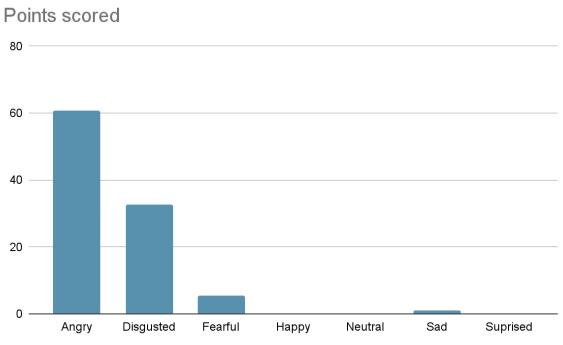
```
test loss, test acc: [1.9080358743667603, 0.5970982313156128]
```

Even though we had a large enough dataset that consisted of 28709 training, 4694 validation and 3590 images, our model did not provide results that can satisfy one of

the most important features we offer to our customers: Reliability. In the experiments we conducted showed that our model was able to capture emotions: happy, neutral and sad, good enough but on other emotions it was not reliable enough to be served. The discrepancy between rates of training and validation data is generally caused by a phenomenon called overfitting and very much applies to our case as well. However, it takes a lot of computational power and time to train a complicated model. Training the model with the GPU's we had in our computers was not enough to achieve SOTA level. The results for some sample tests are provided below:







As you can see even though the model was able to capture happiness in the first photo, in the second and third photo it did not extract emotions. Therefore we

decided that we will use a pretrained model to offer the best possible service to our customers.

4.3.2 Stress Detection

The stress detection architecture works by detecting the eyebrow movements of the interviewees. Related functions and their descriptions can be found as below:

Function	Description
eye_brow_distance(leye,reye)	This function measures the eyebrow distance from eye by the given
normalize_values(points,disp)	This function returns the face and hands landmarks of the interviewees.

4.3.3 Deception Detection

In an online interview, whether the interviewee is lying or not is determined by looking at some actions. These actions can be listed as change in gaze, lip compression, hand covering the face, change in the frequency of the blinking [6].

Function	Description
get_face_relative_area(face)	This function gets the width and height of the face from the frame.
find_face_and_hands(image_original, face_mesh, hands)	This function returns the face and hands landmarks of the interviewees.
is_blinking(face)	This function checks whether the interviewee is blinking or not by looking at the special coordinates of the eyes in his/her face. If it is, returns true, otherwise, returns false.

get_blink_tell(blinks)	This function checks the frequency of blinking. If the interviewee's blinking increases, it returns "Increased blinking". If it decreases, it returns "Decreased blinking".
check_hand_on_face(hands, landmarks, face)	This function checks whether the interviewee's hands are covering his/her face. It detects the face and hands separately and checks them if they overlap or not. If covering, then it returns true, otherwise, it returns false.
detect_gaze_change(avg_gaze)	detect_gaze_change function returns the gaze relative match. If no change in gaze, then it returns 0.

5. 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	● 16.05.2023 ● The user logged in successfully and redirected to

	the company main menu page.

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	● 16.05.2023 ● The user logged in successfully and redirected to the company main menu page.

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	● 16.05.2023

	● The user logged in successfully and redirected to the company main menu page.

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	● 16.05.2023 ● The user logged in successfully and redirected to the company main menu page.

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	• 16.05.2023 • The application should be saved to the database successfully.

Test Case ID	TC_InAI_06
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	• 16.05.2023 • The interview date cannot be a past day (failure in interview creation)

Test Case ID	TC_InAI_07
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	● 16.05.2023 ● An interviewee cannot have overlapping interviews (failure in interview creation)

Test Case ID	TC_InAI_08
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	● 16.05.2023 ● Success in joining the interview

Test Case ID	TC_InAI_09

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	● 16.05.2023 ● Success in joining the interview

Test Case ID	TC_InAI_10
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	● 18.05.2023 ● Save the created company account to the database.

Test Case ID	TC_InAI_11
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	● 18.05.2023 ● Run time exception with “Company already exists” message.

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 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	● 18.05.2023 ● Run time exception with “Company already exists” message.

Test Case ID	TC_InAI_13
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	● 18.05.2023 ● The function returned false.

Test Case ID	TC_InAI_14
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	● 18.05.2023 ● The function returned true. ● Could not test the email sending because testing is done on a mock repository.

Test Case ID	TC_InAI_15
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	● 18.05.2023 ● The function returned false.

Test Case ID	TC_InAI_16
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	● 18.05.2023 ● The password is within rules.

Test Case ID	TC_InAI_17
Test Case Category	Unit Test
Test Case Objective	Successfully creating a non blank password
Precondition	
Procedure	1. In the “CompanyService.java” file, call the generatePassayPassword function
Test Data	
Expected Result	A password with the given rule
Priority	Minor
Date Tested and Result	● 18.05.2023 ● The password is within the rule.

Test Case ID	TC_InAI_18
Test Case Category	Unit Test
Test Case Objective	Successfully delete a company
Precondition	● A valid company account
Procedure	1. In the “CompanyService.java” file, call the deleteByEmail function
Test Data	● A valid company email
Expected Result	The company with the given email is deleted
Priority	Minor
Date Tested and Result	● 19.05.2023 ● The company is deleted

Test Case ID	TC_InAI_19
Test Case Category	Unit Test
Test Case Objective	Fail deleting a company
Precondition	● A valid company account

Procedure	1. In the “CompanyService.java” file, call the deleteByEmail function
Test Data	● An invalid company email
Expected Result	The function must return false
Priority	Minor
Date Tested and Result	● 19.05.2023 ● The function returned false

Test Case ID	TC_InAI_20
Test Case Category	Unit Test
Test Case Objective	Get a company by its name
Precondition	● A valid company account
Procedure	1. In the “CompanyService.java” file, call the getByNameSuccessful function
Test Data	● A valid company name
Expected Result	The function return the company with the given name
Priority	Minor
Date Tested and Result	● 19.05.2023 ● The function returned the correct company

Test Case ID	TC_InAI_21
Test Case Category	Unit Test
Test Case Objective	Fail getting a company by its name
Precondition	● A valid company account
Procedure	1. In the “CompanyService.java” file, call the deleteByEmail function
Test Data	● An invalid company name
Expected Result	The function must return null
Priority	Minor
Date Tested and Result	● 19.05.2023 ● The function returned null

Test Case ID	TC_InAI_22
Test Case Category	Unit Test
Test Case Objective	Successfully join 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 correct interview
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function returned the correct interview

Test Case ID	TC_InAI_23
Test Case Category	Unit Test
Test Case Objective	Fail joining to a meeting as an interviewee because of wrong password
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	● 19.05.2023 ● The function returned null

Test Case ID	TC_InAI_24

Test Case Category	Unit Test
Test Case Objective	Fail in joining to a meeting as an interviewee because of nonexistent interview
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	● 19.05.2023 ● The function returned null

Test Case ID	TC_InAI_25
Test Case Category	Unit Test
Test Case Objective	Successfully check whether the interviewee joined the meeting
Precondition	● A valid interview id ● A valid interviewee
Procedure	1. In the “IntervieweeService.java” file, call the “isJoined” function
Test Data	● A valid interview id
Expected Result	The function must return true or false depending on whether the interviewee joined or not
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function returned accurately

Test Case ID	TC_InAI_26
Test Case Category	Unit Test
Test Case Objective	Failed checking whether the interviewee joined the meeting because of nonexistent interview

Precondition	• An invalid interview id • A valid interviewee
Procedure	1. In the “IntervieweeService.java” file, call the “isJoined” function
Test Data	• An invalid interview id
Expected Result	The function must throw an exception
Priority	Major
Date Tested and Result	• 19.05.2023 • The function throwed exception

Test Case ID	TC_InAI_27
Test Case Category	Unit Test
Test Case Objective	Save an interviewee successfully to the database
Precondition	• A valid interviewee id
Procedure	1. In the “IntervieweeService.java” file, call the “save” function
Test Data	• A valid interviewee id
Expected Result	The function must save the interviewee to the database
Priority	Major
Date Tested and Result	• 19.05.2023 • The function saved the interviewee to the database

Test Case ID	TC_InAI_28
Test Case Category	Unit Test
Test Case Objective	Do not save an already existing interviewee to the database
Precondition	• A valid interviewee id
Procedure	1. In the “IntervieweeService.java” file, call the “save” function
Test Data	• An already taken interviewee id
Expected Result	The function must not save the interviewee to the

	database
Priority	Major
Date Tested and Result	• 19.05.2023 • The function did not save the interviewee to the database

Test Case ID	TC_InAI_29
Test Case Category	Unit Test
Test Case Objective	Successfully get all interviewees in the database
Precondition	
Procedure	1. In the “IntervieweeService.java” file, call the “getAll” function
Test Data	
Expected Result	The function must return every interviewee in the database
Priority	Major
Date Tested and Result	• 19.05.2023 • The function returned every interviewee in the database

Test Case ID	TC_InAI_30
Test Case Category	Unit Test
Test Case Objective	Successfully get and interviewee by its id
Precondition	• A valid interviewee
Procedure	1. In the “IntervieweeService.java” file, call the “getById” function
Test Data	• A valid interviewee id
Expected Result	The function must return the specific interviewee
Priority	Major
Date Tested and Result	• 19.05.2023 • The function returned the specific interviewee

Test Case ID	TC_InAI_31
Test Case Category	Unit Test
Test Case Objective	Fail in getting a nonexistent interviewee
Precondition	
Procedure	1. In the “IntervieweeService.java” file, call the “getById” function
Test Data	● An invalid interviewee id
Expected Result	The function must throw an exception
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function threw an exception

Test Case ID	TC_InAI_32
Test Case Category	Unit Test
Test Case Objective	Successfully get an interviewee by its email
Precondition	● A valid interviewee account
Procedure	1. In the “IntervieweeService.java” file, call the “getByEmail” function
Test Data	● A valid interviewee email
Expected Result	The function must return the specific interviewee
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function returned the specific interviewee

Test Case ID	TC_InAI_33
Test Case Category	Unit Test
Test Case Objective	Failure in getting a nonexistent interviewee by its email
Precondition	
Procedure	1. In the “IntervieweeService.java” file, call the

	“getByEmail” function
Test Data	• An invalid interviewee email
Expected Result	The function must throw an exception
Priority	Major
Date Tested and Result	• 19.05.2023 • The function threw an exception

Test Case ID	TC_InAI_34
Test Case Category	Unit Test
Test Case Objective	Successfully delete an interviewee by its email
Precondition	• An interviewee
Procedure	1. In the “IntervieweeService.java” file, call the “deleteByEmail” function
Test Data	• A valid interviewee email
Expected Result	The function must return true
Priority	Major
Date Tested and Result	• 19.05.2023 • The function returned true

Test Case ID	TC_InAI_35
Test Case Category	Unit Test
Test Case Objective	Failure in deleting a nonexistent interviewee by its email
Precondition	
Procedure	1. In the “IntervieweeService.java” file, call the “deleteByEmail” function
Test Data	• An invalid interviewee email
Expected Result	The function must return false
Priority	Major
Date Tested and Result	• 19.05.2023 • The function returned false

Test Case ID	TC_InAI_36
Test Case Category	Unit Test
Test Case Objective	Failure in deleting a nonexistent interviewee by its email
Precondition	
Procedure	1. In the “IntervieweeService.java” file, call the “deleteByEmail” function
Test Data	● An invalid interviewee email
Expected Result	The function must return false
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function returned false

Test Case ID	TC_InAI_37
Test Case Category	Unit Test
Test Case Objective	Successfully create an interview with a nonexistent interviewee
Precondition	● A valid interviewer id ● Interviewer does not have any interview at the given time
Procedure	1. In the “InterviewService.java” file, call the “create” function
Test Data	● A valid interviewee email ● A valid interviewee first name ● A valid interviewee last name ● A valid applied position ● A valid CV
Expected Result	The function must save data hangar, analysis data, interview and interviewee objects to the database
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function saved all of the given objects

Test Case ID	TC_InAI_38
Test Case Category	Unit Test
Test Case Objective	Successfully create an interview with an existent interviewee
Precondition	● A valid interviewer id ● Interviewer does not have any interview at the given time ● Interviewee does not have any interview at the given time
Procedure	1. In the “InterviewService.java” file, call the “create” function
Test Data	● A valid interviewee email ● A valid interviewee first name ● A valid interviewee last name ● A valid applied position ● A valid CV
Expected Result	The function must save data hangar, analysis data and interview objects to the database
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function saved all of the given objects

Test Case ID	TC_InAI_39
Test Case Category	Unit Test
Test Case Objective	Failure in creating an interview with an existent interviewee that has interview at the given time
Precondition	● A valid interviewer id ● Interviewer does not have any interview at the given time ● Interviewee has an interview at the given time
Procedure	1. In the “InterviewService.java” file, call the “create” function
Test Data	● A valid interviewee email ● A valid interviewee first name ● A valid interviewee last name ● A valid applied position ● A valid CV
Expected Result	The function must throw an exception

Priority	Major
Date Tested and Result	• 19.05.2023 • The function threw an exception

Test Case ID	TC_InAI_40
Test Case Category	Unit Test
Test Case Objective	Failure in creating an interview with an existent interviewee because of interviewer has an interview at the given time
Precondition	• A valid interviewer id • Interviewer has an interview at the given time
Procedure	1. In the “InterviewService.java” file, call the “create” function
Test Data	• A valid interviewee email • A valid interviewee first name • A valid interviewee last name • A valid applied position • A valid CV
Expected Result	The function must throw an exception
Priority	Major
Date Tested and Result	• 19.05.2023 • The function threw an exception

Test Case ID	TC_InAI_41
Test Case Category	Unit Test
Test Case Objective	Failure in creating an interview with a nonexistent interviewee because of interviewer has an interview at the given time
Precondition	• A valid interviewer id • Interviewer has an interview at the given time
Procedure	1. In the “InterviewService.java” file, call the “create” function
Test Data	• A valid interviewee email • A valid interviewee first name • A valid interviewee last name • A valid applied position • A valid CV

Expected Result	The function must throw an exception
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function threw an exception

Test Case ID	TC_InAI_42
Test Case Category	Unit Test
Test Case Objective	Successfully get all interviews
Precondition	
Procedure	1. In the “InterviewService.java” file, call the “getAll” function
Test Data	
Expected Result	The function must return all interviews in the database
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function returned every interviews in the database

Test Case ID	TC_InAI_43
Test Case Category	Unit Test
Test Case Objective	Successfully get an interview by its id
Precondition	● A valid interview
Procedure	1. In the “InterviewService.java” file, call the “getById” function
Test Data	● A valid interview id
Expected Result	The function must return the specific interview from the database
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function returned the specific interview

Test Case ID	TC_InAI_44
Test Case Category	Unit Test
Test Case Objective	Failure in getting a nonexistent interview
Precondition	
Procedure	1. In the “InterviewService.java” file, call the “getById” function
Test Data	● An invalid interview id
Expected Result	The function must throw an exception
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function threw an exception

Test Case ID	TC_InAI_45
Test Case Category	Unit Test
Test Case Objective	Successfully get interviews of a specific interviewer
Precondition	● A valid interviewer
Procedure	1. In the “InterviewService.java” file, call the “getInterviewsByInterviewerId” function
Test Data	● A valid interviewer id
Expected Result	The function must return all of the interviews of the interviewer
Priority	Major
Date Tested and Result	● 19.05.2023 ● The function returned all of the interviews of the interviewer

Test Case ID	TC_InAI_46
Test Case Category	Unit Test
Test Case Objective	Successfully get admin by its email

Precondition	● A valid admin
Procedure	1. In the “AdminService.java” file, call the “getByEmail” function
Test Data	● A valid admin email
Expected Result	The function must return the correct admin
Priority	Minor
Date Tested and Result	● 19.05.2023 ● The function returned the specific admin

Test Case ID	TC_InAI_47
Test Case Category	Unit Test
Test Case Objective	Failure in getting a nonexistent admin by its email
Precondition	
Procedure	1. In the “AdminService.java” file, call the “getByEmail” function
Test Data	● An invalid admin email
Expected Result	The function must return null
Priority	Minor
Date Tested and Result	● 19.05.2023 ● The function returned null

Test Case ID	TC_InAI_48
Test Case Category	Unit Test
Test Case Objective	Successfully login as an admin
Precondition	● A valid admin
Procedure	1. In the “AdminService.java” file, call the “login” function
Test Data	● A valid admin email ● A valid admin password
Expected Result	The function must return the correct admin

Priority	Critical
Date Tested and Result	• 19.05.2023 • The function returned the specific admin

Test Case ID	TC_InAI_49
Test Case Category	Unit Test
Test Case Objective	Failure in signing in as an admin because of wrong email
Precondition	• A valid admin
Procedure	1. In the “AdminService.java” file, call the “login” function
Test Data	• An invalid admin email • A valid admin password
Expected Result	The function must return null
Priority	Critical
Date Tested and Result	• 19.05.2023 • The function returned null

Test Case ID	TC_InAI_50
Test Case Category	Unit Test
Test Case Objective	Failure in signing in as an admin because of wrong password
Precondition	• A valid admin
Procedure	1. In the “AdminService.java” file, call the “login” function
Test Data	• A valid admin email • An invalid admin password
Expected Result	The function must return null
Priority	Critical
Date Tested and Result	• 19.05.2023 • The function returned null

6. Maintenance Plan and Details

Since this project is intended to be put on the market, it is important for the project to be kept updated in terms of both the future improvements and the libraries in the technologies used. These can be investigated one by one in the following items:

6.1 Scalability and Future Enhancements:

In terms of future scale and potential improvements, it is important that this project remains feasible for the long term. In the maintenance plan, we will consider future user growth, rising user demand and possible implementation of features. It may include adding more servers, improving the codebase and taking user feedback into account in order to continuously improve the system.

6.2 User Support and Training:

In order for companies to successfully adopt our website, it is essential that they receive user support and training. A comprehensive user support system should be created which will include user documentation, FAQs and help desk with a view to make it easy for users to use our application. In order to make companies more familiar with its features, training sessions can be held showing them to use interview data in the most effective way.

6.3 Bug Fixing and Troubleshooting:

The maintenance plan should include the provision of bug correction and troubleshooting. If users use Interly.ai, there may be problems that needed to be solved immediately. An effective system for tracking and addressing bugs is needed to facilitate the rapid identification, investigation and resolution of problems.

6.4 Infrastructure Maintenance:

It is important that the underlying infrastructure is maintained in order to ensure a straight implementation for the project. Regular maintenance of the servers, database optimization and monitoring of system performance are also part of this. Regular updates of the infrastructure, via security patches and software upgrades, should be carried out to mitigate potential vulnerabilities.

6.5 Model Training and Optimization:

Ongoing training and optimization of deep learning models is needed in view of their central role in our project's functionality. To improve accuracy and performance, reconfiguring models that use new dataset will be a significant process. Our maintenance plan should schedule updating the models in the current system and determine criteria for their redeployment to keep them functioning all the time.

7. Other Project Elements

7.1 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 [7]. 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.

7.2 Ethics and Professional Responsibilities

Interly.ai aims to help both companies and candidates by automatizing and clarifying hiring processes by utilizing the latest innovations in the machine learning field. While serving our great range of functionality to everyone that wants to benefit from the interly.ai system, we are well aware that the data processed inside the system is very personal, sensitive and should never be leaked. As developers of such a vulnerable application we will always try to prioritize protecting the privacy and safety of our customers' sensitive information. To ensure our customers never have doubts and feel nervous about using our product, we will always make decisions ethically, responsibly and fairly both in design and implementation phase.

Another ethical issue about projects making suggestions about real individuals characteristics and thus recommending hiring managers into a certain direction, even though the last decision is always up to the employees, is treating everyone equally and fairly. Any kind of discrimination could change the lives of thousands of individuals and can affect a company's productivity. As engineers we promise to treat every candidate equally no matter what their race, gender, ethnicity, religion or culture is. Especially, when training our AI model, we tried to train a model that has no biases against any type of minority.

7.3 Teamwork Details

7.3.1 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 continued 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. The completed tasks with their corresponding assignees are as follows:

Task No	Title	Leader	Members	Finish Date
1	Deep Learning: Emotion Recognition Implementation Enhancements	Murat Furkan Uğurlu	Furkan Turunç, Osman Semih Tiryaki, Erhan Er, Arda Serim	April 25
2	Deep Learning: Stress Detection	Osman Semih	Murat Furkan Uğurlu, Furkan Turunç, Erhan	April 18

		Tiryaki	Er, Arda Serim	
3	Performance Tuning: Improvements on Real-Time Graphs in Interview	Arda Serim	Murat Furkan Uğurlu, Erhan Er	April 20
4	Frontend: Changes and Arrangements	Erhan Er	Arda Serim, Murat Furkan Uğurlu	May 13
5	Backend: Changes and Arrangements	Arda Serim	Osman Semih Tiryaki, Furkan Turunç	May 15
6	Prepare Executable	Osman Semih Tiryaki	Furkan Turunç, Murat Furkan Uğurlu	May 18

7.3.2 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.

7.3.3 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.

7.3.4 Meeting Objectives

As of the time of writing this report, most of the proposed functionality has been implemented and requirements we managed to fulfill are:

7.4 New Knowledge Acquired and Applied

In the process of developing a hybrid application that has both a website part and a desktop plugin that cooperates in the local machine of interviewees requires extra learning for various aspects of application and web development. Also the plugins that work behind the scenes consist of different machine learning models working concurrently and collaboratively to provide the services we offer to our customers. At the start of the project, the knowledge we had was not enough to complete the services and deploy the end product. Therefore, we need to enhance our knowledge with these 3 main methods:

- Literature Review
- Online Learning
- Hands-on Experience

Using these methods we enhanced our knowledge on various fields that could possibly arise in a development lifecycle of an app. Following examples are the most important topics we have acquired during this Senior project:

- Web Development
- Software Project Design
- Machine Learning
- Integrating AI models into applications
- Software Architecture Planning

We had previous knowledge about web development and software architecture planning but it was not enough and we had to refresh and improve our knowledge about those two topics.

As for machine learning, designing an architecture with these different new pieces and integrating those pieces into a web based video conferencing application was completely new for us. Thus we had to learn from scratch about those topics and use the theoretical knowledge to build the application.

8. Conclusion and Future Work

In conclusion, Interly.ai project successfully achieved its objective of developing a system to analyze interviewees' using deep learning. Interly.ai, with what it offers beyond current technologies used in recruitment processes, revolutionizes the interview process and enhances decision making in the hiring process.

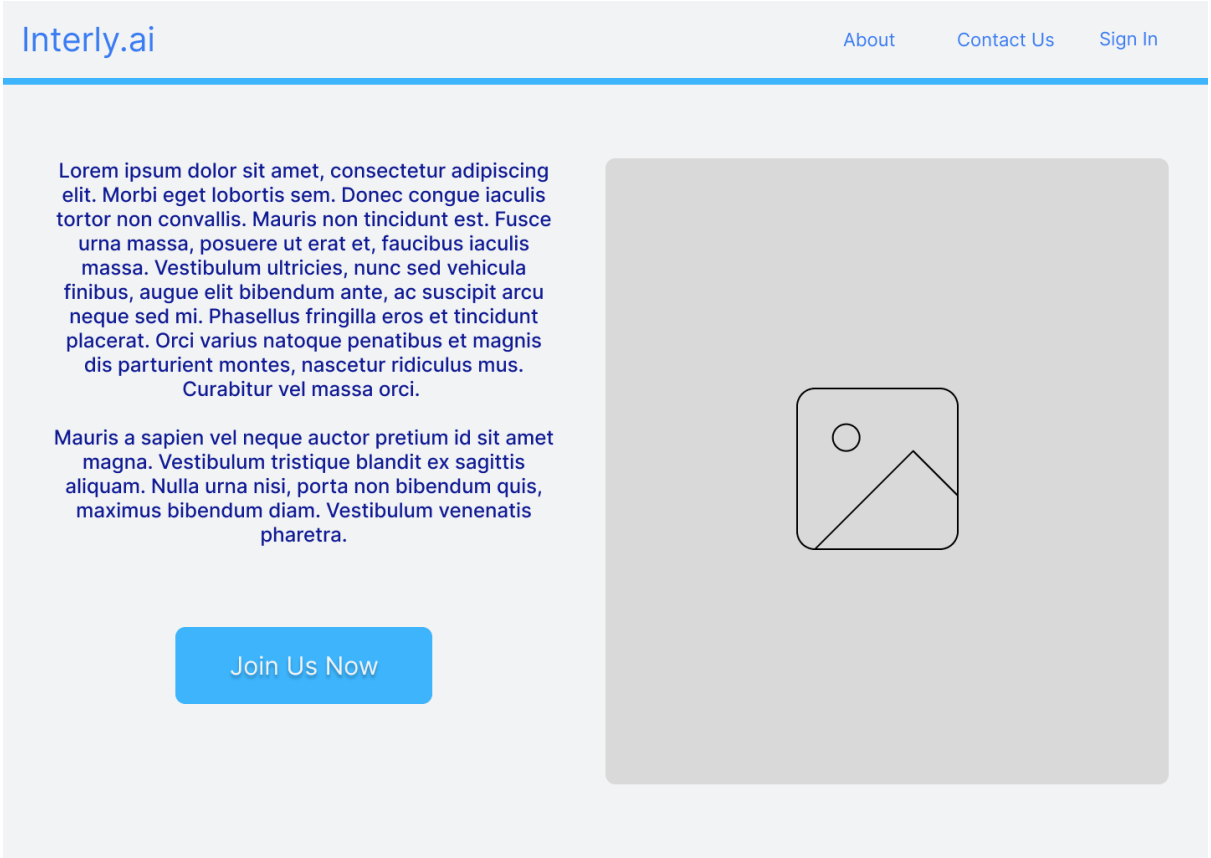
We used cutting-edge deep learning models and effective data analysis techniques throughout the project to precisely assess and understand the emotions, stress levels and deception of interviewees. The system's effectiveness was thoroughly assessed, and it showed promising results in properly identifying emotional states and degrees of stress during interviews.

With the launch of the website, businesses now have a centralized location to access and use the vast amount of interview data acquired by the system. Employing this platform allows businesses to better analyze applicant responses and match recruiting choices with desired corporate culture and values.

User Manual

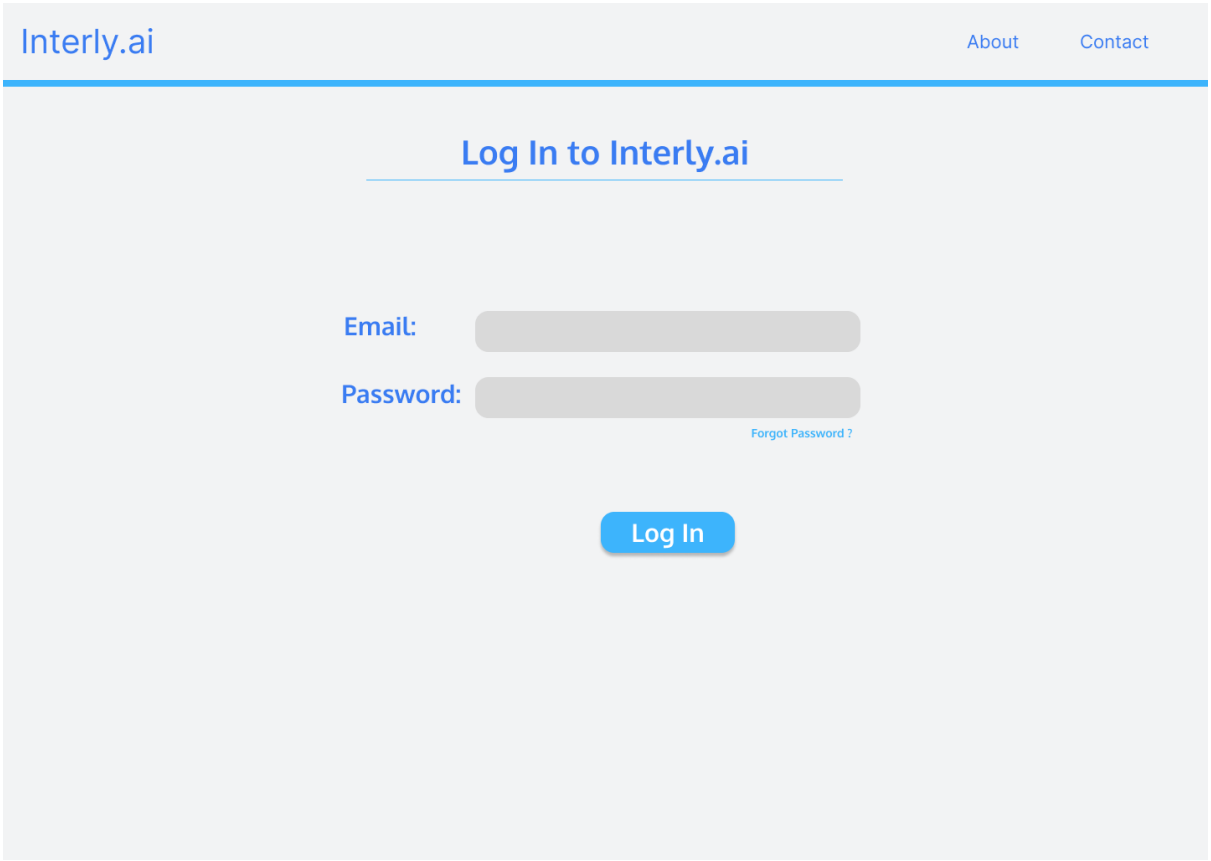
Description of Pages

Landing Page



This is the landing page. The landing page contains a brief description of the application. From the navbar, visitors can access the sign in page, about page and contact with us. If the visitor clicks the join us now button, they can access the company approve form page where they will enter the required information of their company to apply.

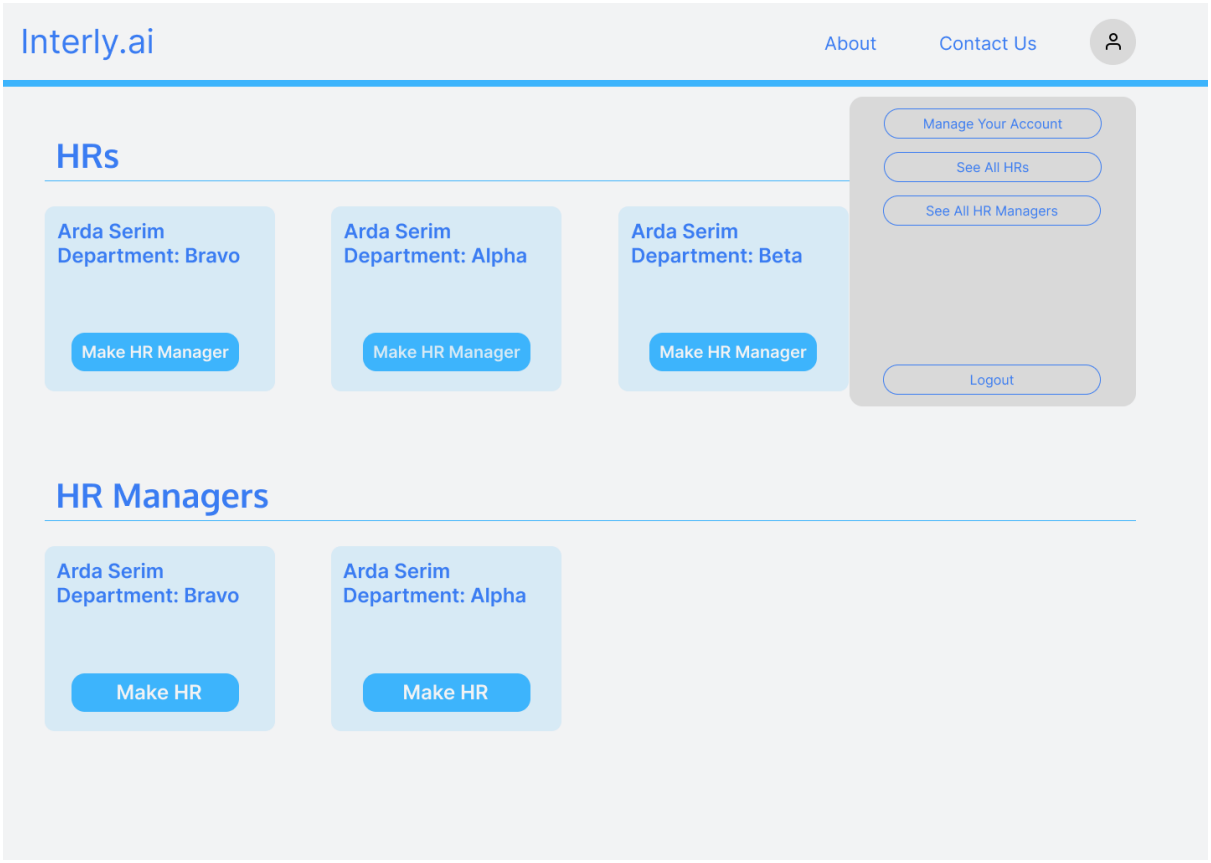
Login Page



This is the sign in page. This page will be the same for all of the users (Admins, Companies, Interviewers and HRMs). Users will enter their emails and passwords. Account types will be decided on the server side after the user clicks the Login button.

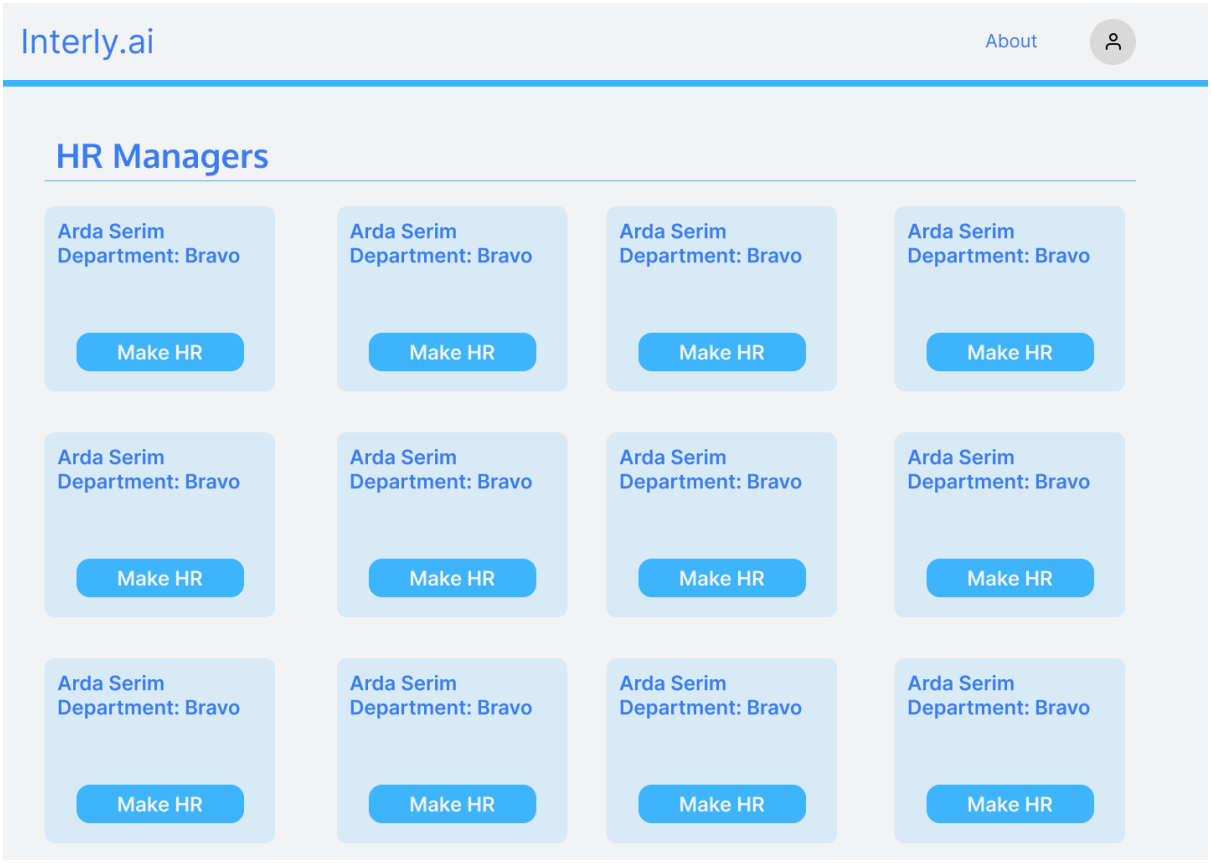
Company

Company - Main Menu



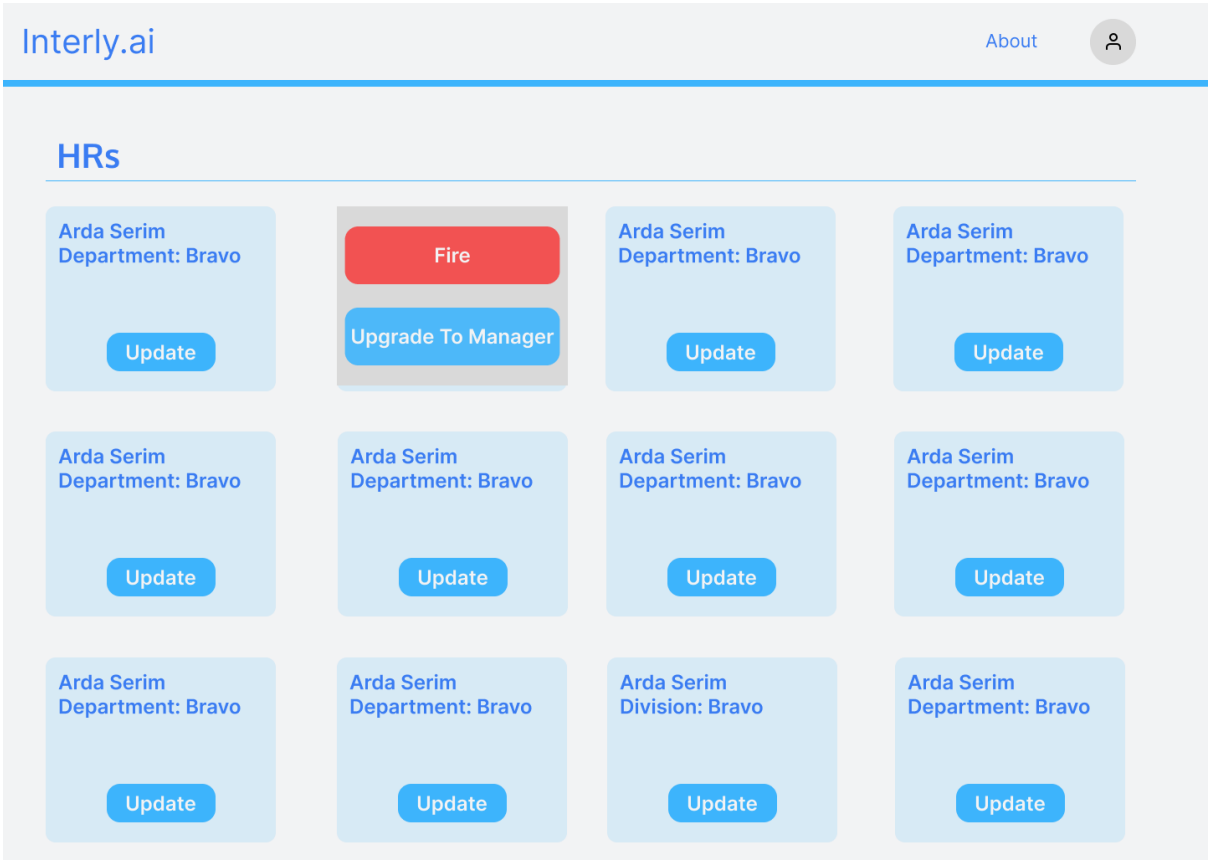
This is a more general page for a company. The information of two previous pages can be seen here also in a more organized and less detailed way. However, the employer operations can also be performed here by clicking the update button which means the functionality is still preserved. Also from the navbar, List of HR's and List of Manager's can be reached to see all of them at the same time. Also by clicking the manage your account, button company settings page can be viewed.

List of HR Managers - Company:



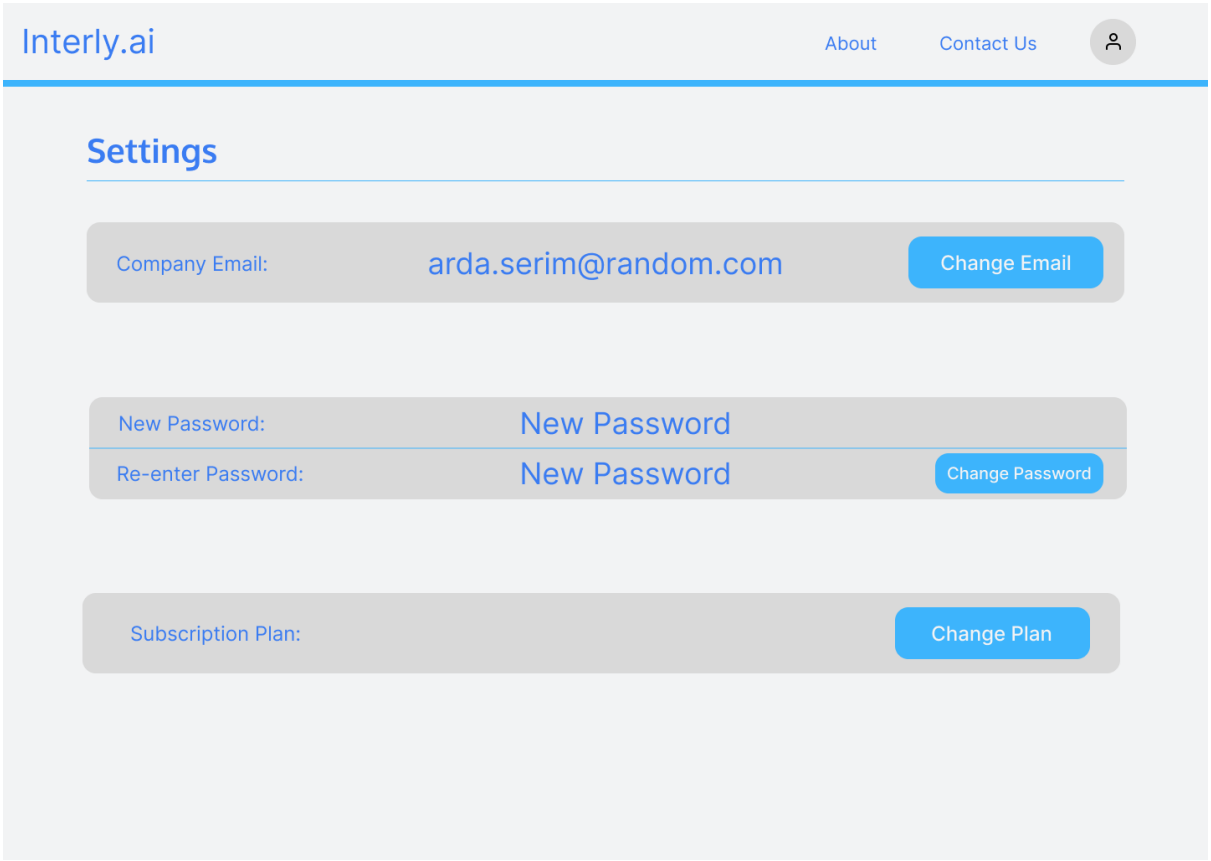
This is the main page for controlling operations about HR managers. Only very privileged users like CEO's or general managers who are very high in the company hierarchy will have access to company accounts and view this page. From here, general managers can review all HR managers, a brief information card about their name and department. Also the managers with access to company accounts can update the status of any HR Manager currently working in the company by just clicking the update button. Currently there are two options planned for an HR manager. A CEO can directly fire an HR manager by clicking the fire button. Alternatively, he/ she can change the role of an HR Manager to a normal HR and the system will automatically change the roles and responsibilities of that user with one click to Downgrade to HR.

List of HRs - Company:



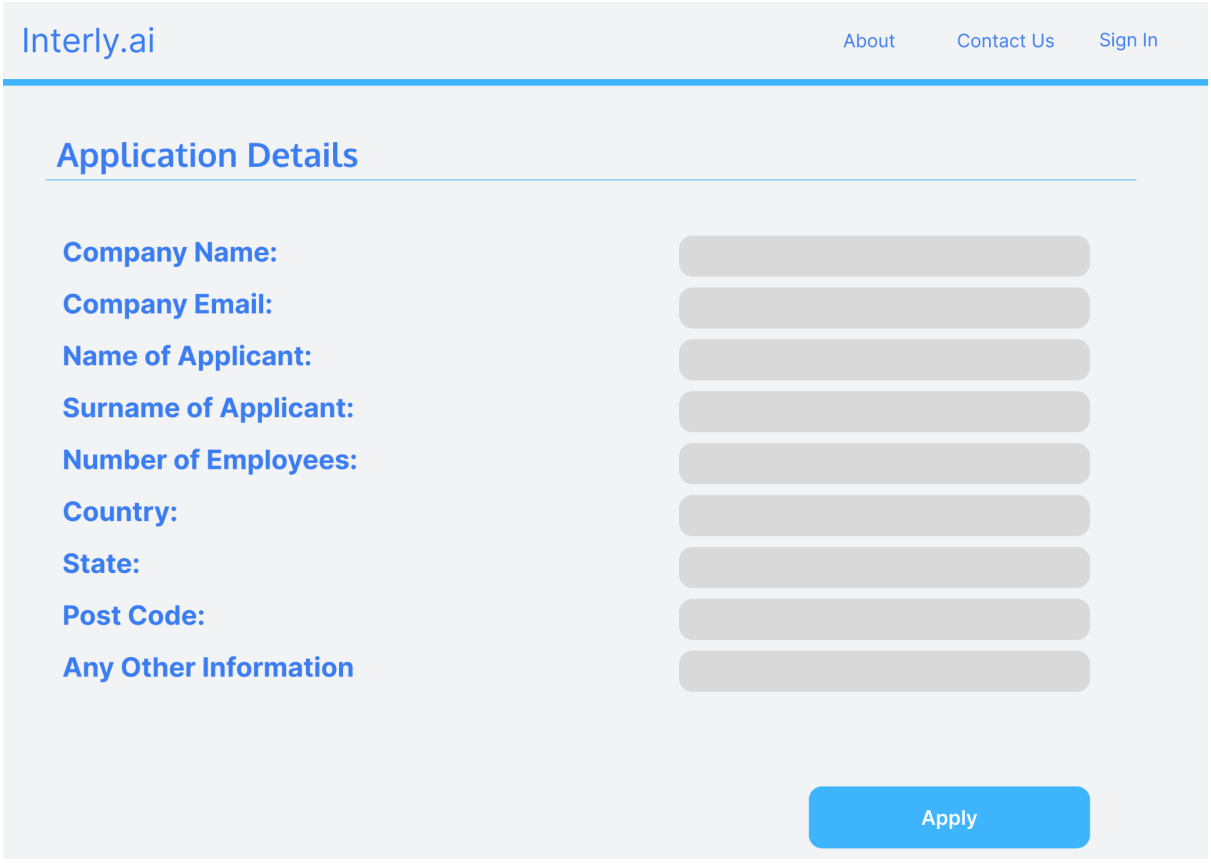
Similar to the previous page, the administrators of the company can handle HR operations in a very user friendly way. They can see all human resources workers currently working in the company and perform operations about their role. After clicking update, they can easily disconnect the role of that HR from the company and refrain from doing anything in the name of the company. Also upgrading an HR could be easily done from this page by the general managers via clicking the Upgrade to Manager button. System will automatically change the role and privileges of that user and update his/her role on other pages.

Company Settings:



This is the account settings page for a company account. The very top directors of the company can access this page. Company mail or password change requests can be done from here by clicking the change email or change password buttons. Since this is a delicate process, the changes will apply only after the request is coming from valid and trustworthy sources. Changing the subscription package will be handled from this page, by clicking the change plan button.

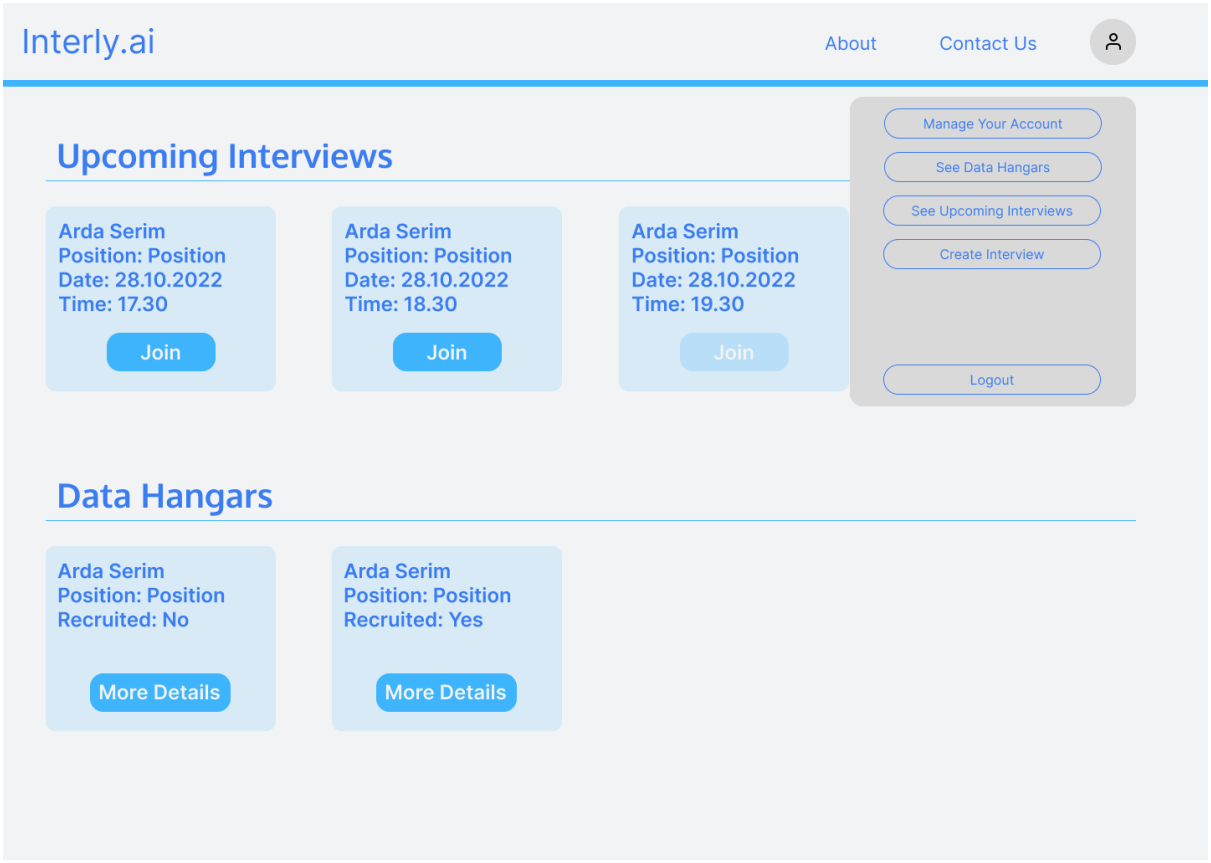
Company Registration Form:



This page is used by companies that want to use Interly.ai. Companies need to fill this form to be able to use our system. After they fill the form and press the Apply button, Admins will be notified. Only if admin approves companies registration, will companies be able to use Interly.ai.

HR

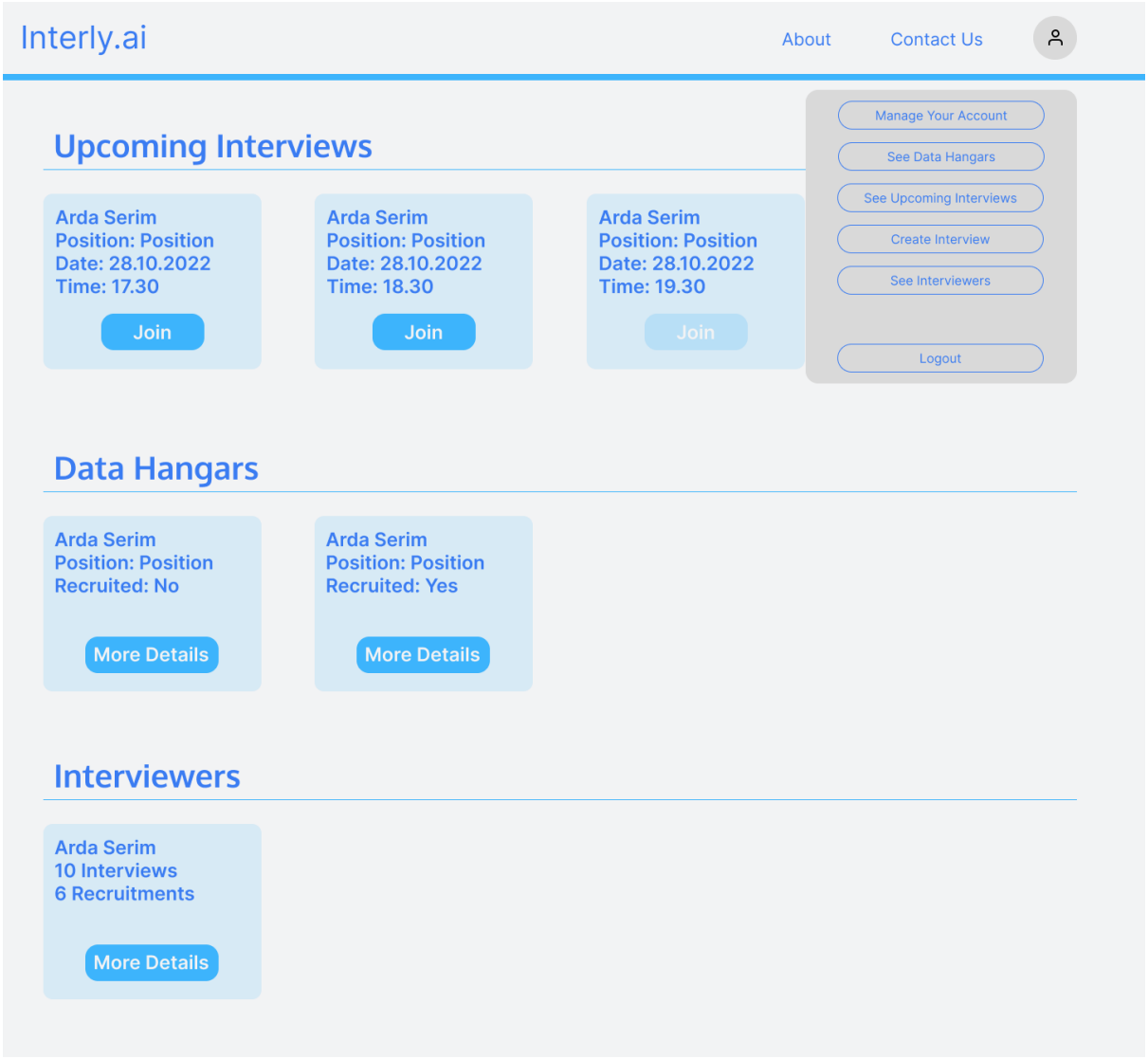
HR Main Page:



This page is the main page of an interviewer and it indicates the upcoming interviews that the interviewer will attend. Each interview is represented separately and shows the name of the candidate, the position that is applied by a candidate of the corresponding interview along with the date and time when the interview is scheduled. The interviewer can join an interview by clicking the button named Join, which is located at the bottom of an interview component. At the bottom of this page, there is a part that shows Data Hangars which is the analysis data of an interview that the interviewer attended before. Each hangar shows the brief information including the name of the candidate, the position s/he applied, and the result of the interview and more information can be viewed by clicking the button named More Details. An interviewer has other options on the main page. S/he can manage his/her account and create interviews as well.

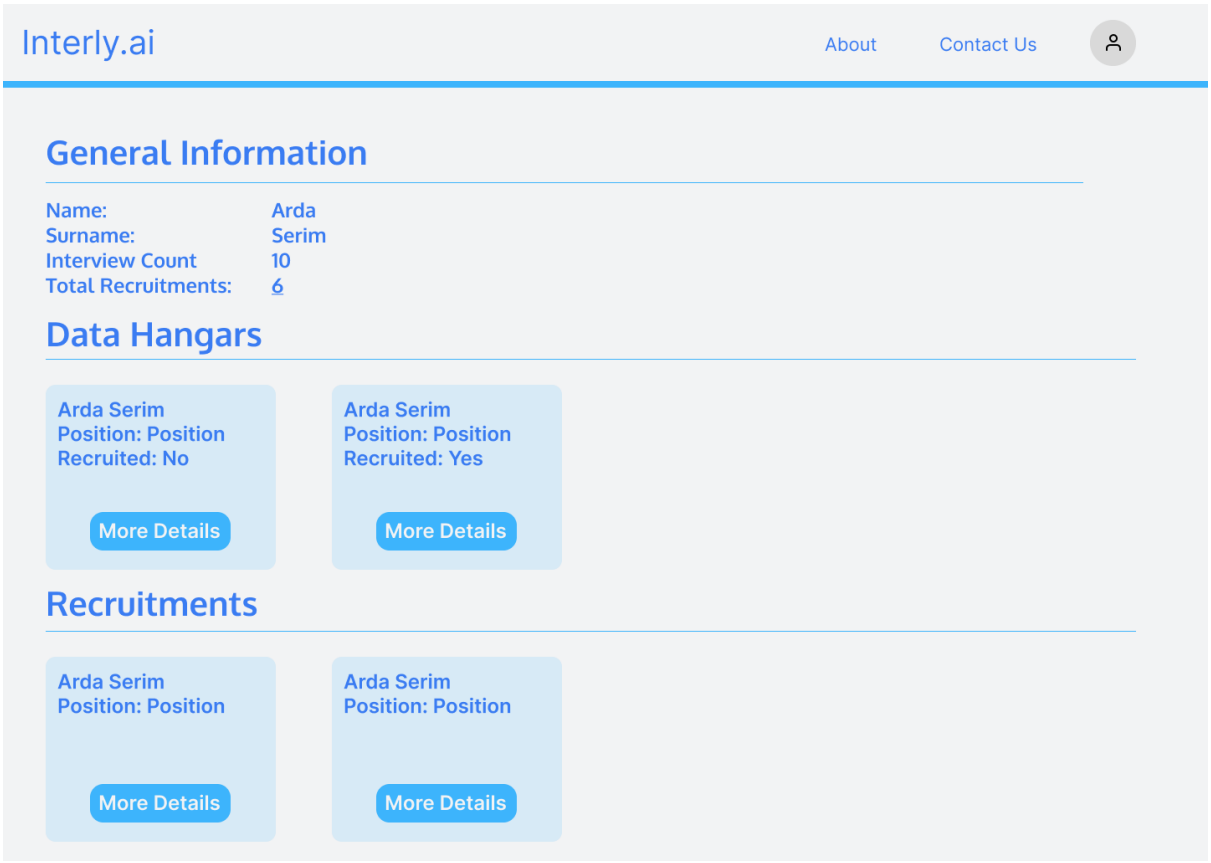
HR Manager

HR Manager Main Page



This is the main page of HR Managers. They can see the upcoming interviews on the screen and join them. If the interview date is close, the join button will become active. Otherwise, it will stay inactive. They can also see the data hangars where analysis of the interviews are held. In addition to that, they can also see the interviewers in the company and a simple report about them which tells the number of interviews that person did and how many of them resulted in recruitment. From the navbar, the user can access all upcoming interviews, data hangars and interviewers and settings and the user can also logout from the navbar. The user can also access the create interview page.

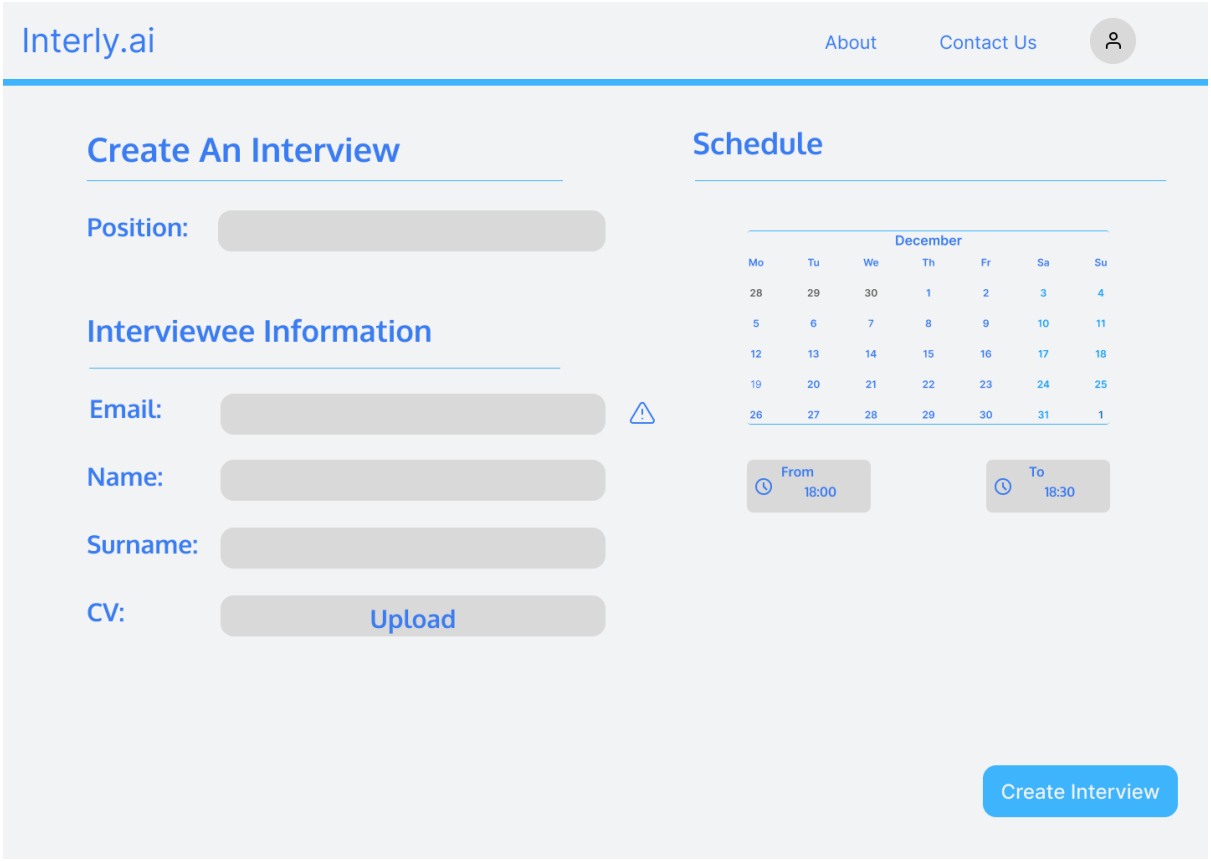
Interviewer Detailed Page



This page is the interviewer's detailed page and s/he can see the number of interviews s/he attended and total recruitments of these interviews. In this page, the interviewer can see the data hangars and the corresponding recruitments. To get more information about the data hangars or interview results, the admin can click the button named More Details.

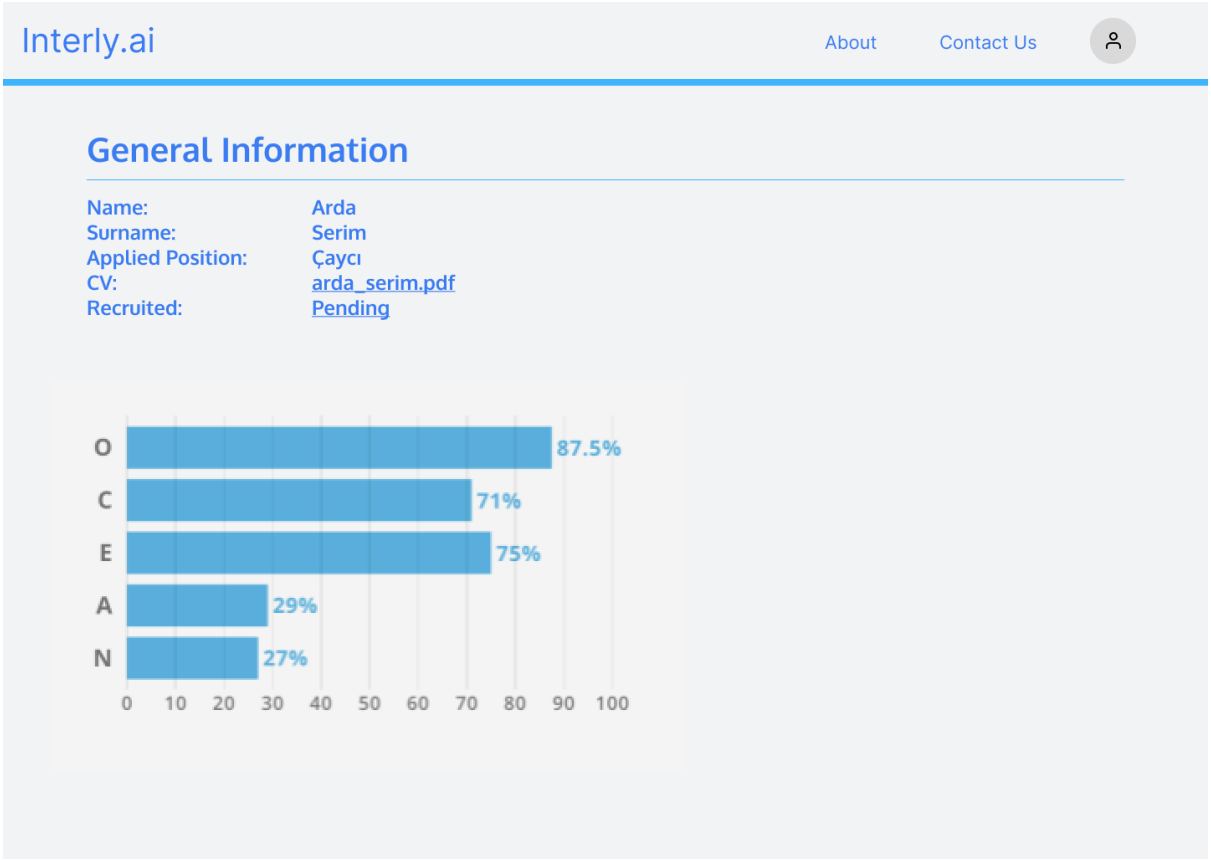
HR & HR Manager

Interview Creation



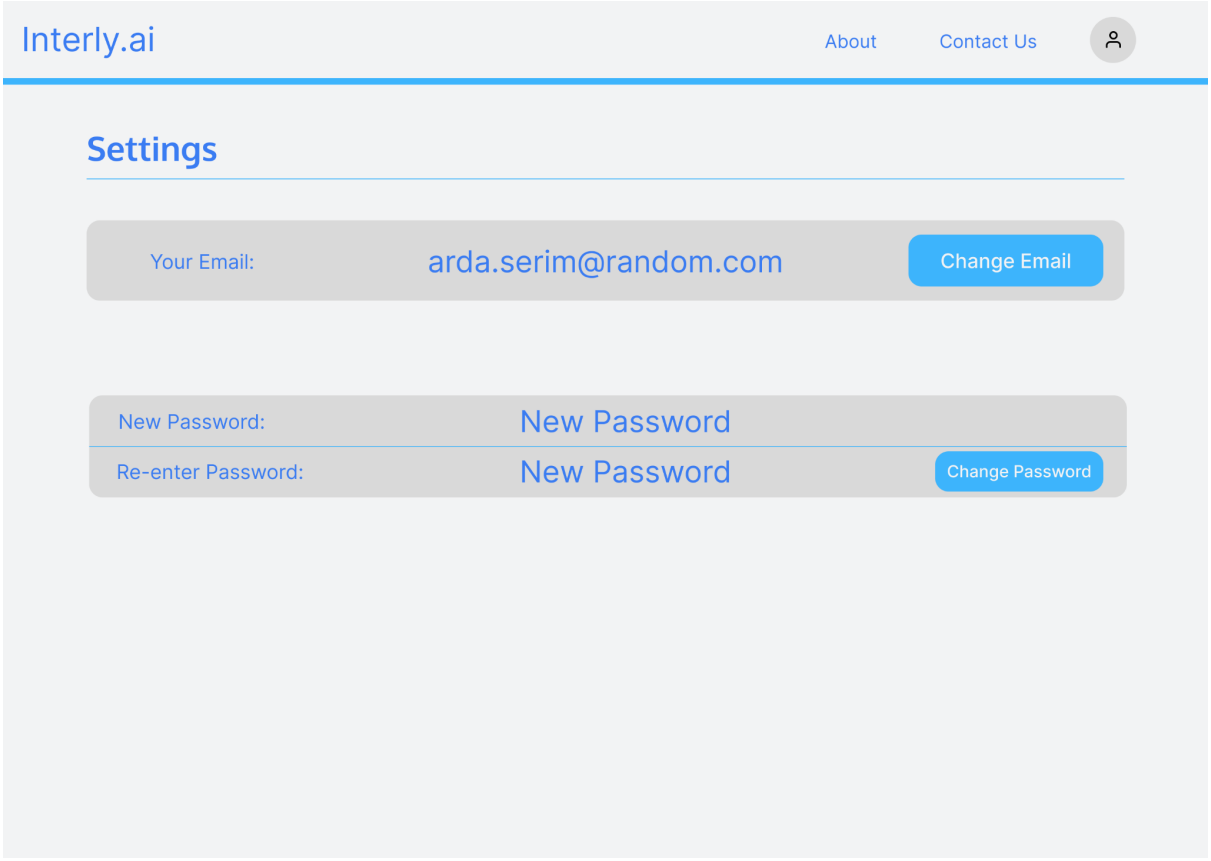
The interview creation screen can be seen by HR Managers and interviewers. The position of the job must be entered. If the email is available in the database which means the interviewee has already done an interview with the company, the other information of the interviewee will be filled automatically. In addition to that, the date and the time can be adjusted on the calendar and time picker. After all necessary information is filled, the interviewer can create the interview.

Data Hangar



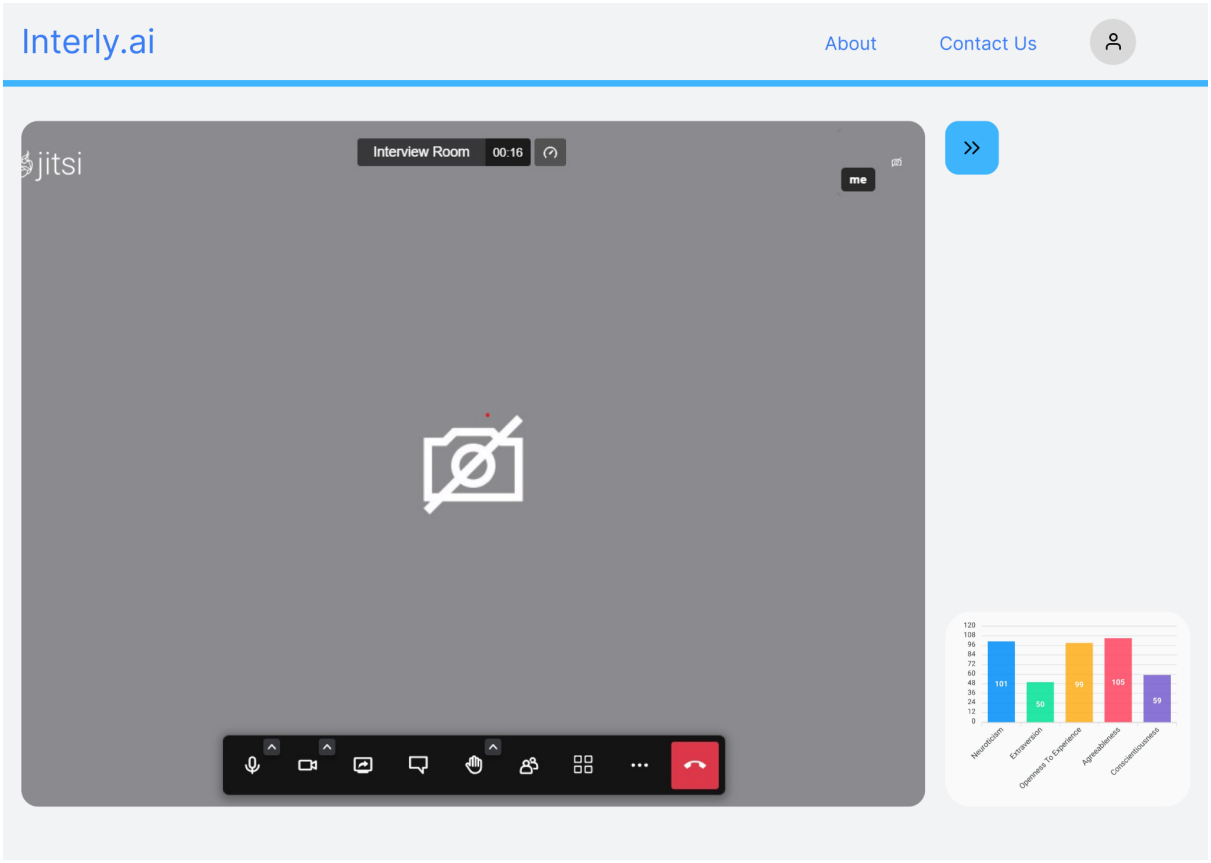
Data hangars are the places where the analysis of the interviews are held. The results are shown bottom with graphics. In addition to that, the general information about the interviewee can be seen on the page and also the information of recruitment can be seen. If the interviewee was rejected, the recruited status will be updated to rejected or no, if it was accepted, the status will be updated to yes or accepted.

Settings Page for HR and HR Manager



This page is the settings page for both interviewers and human resource managers. Users of both user types can change their emails and passwords from this page.

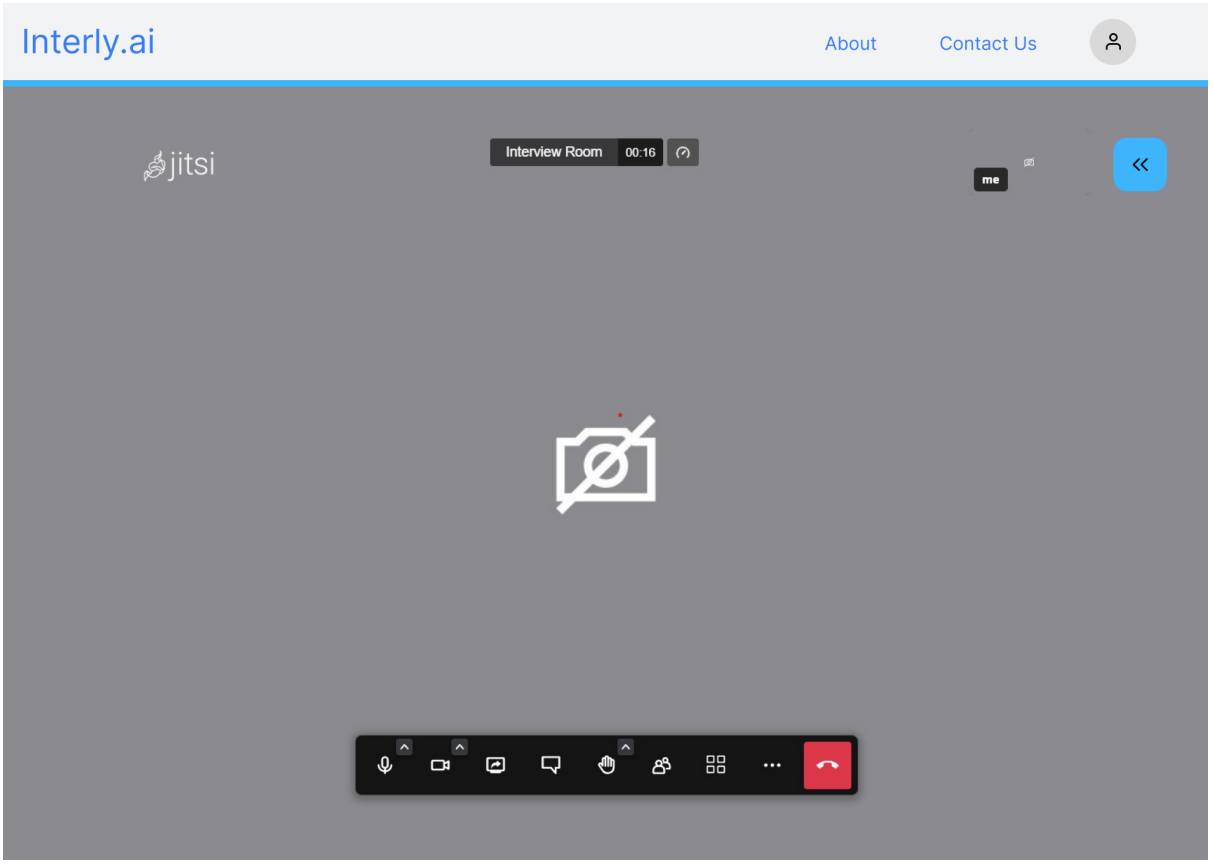
Meeting Page with Analysis Data Shown



This page is almost the same as the previous page. The only difference is there is analysis data of interviewee, which is calculated and shown in real time.

HR & HR Manager & Interviewee

Meeting Page:



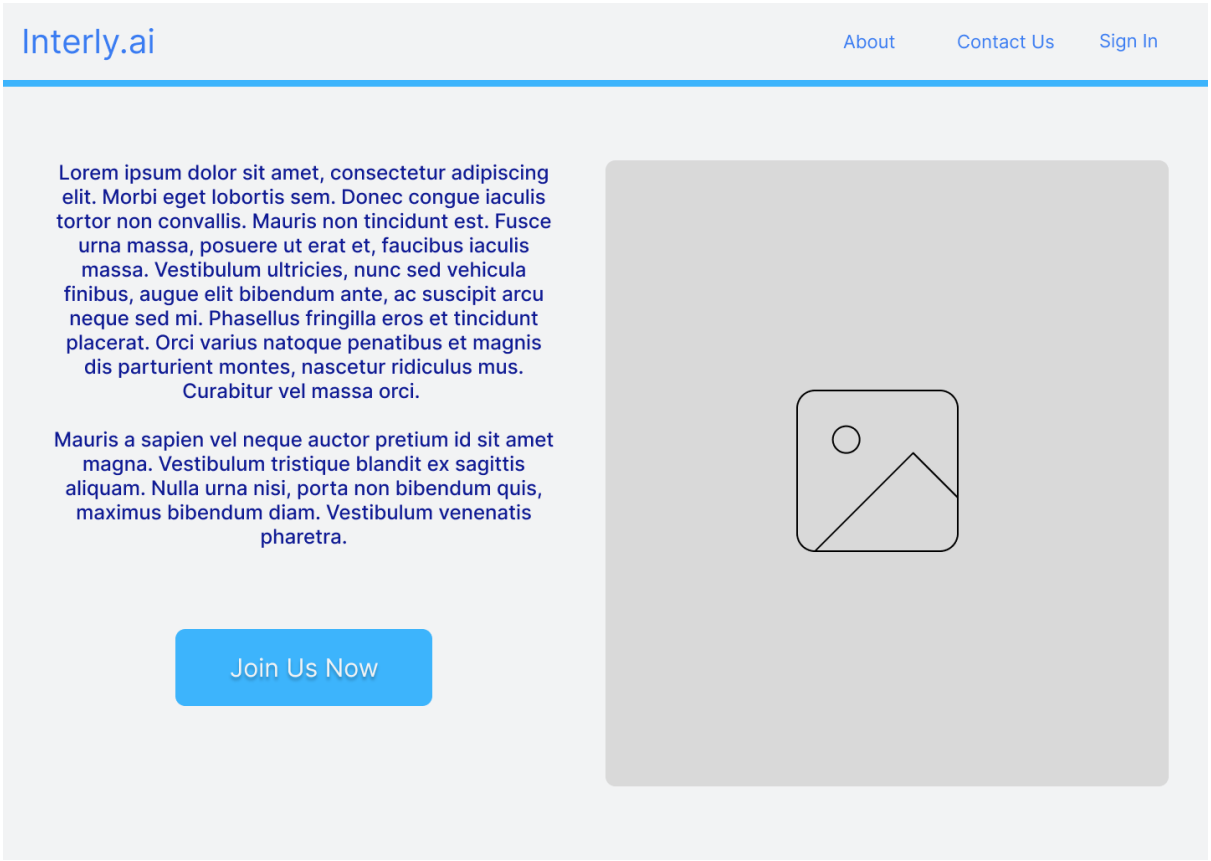
This page is the meeting page. Both interviewers and interviewees can see this page. They can mute/unmute, share screen, turn on/off their camera and leave the meeting. And on the upper right side of the page there is a little blue arrow button. This button is only available to the interviewers. This button makes the interviewer able to see the interviewee's real time analysis data.

Sample Scenarios

Creating Interview

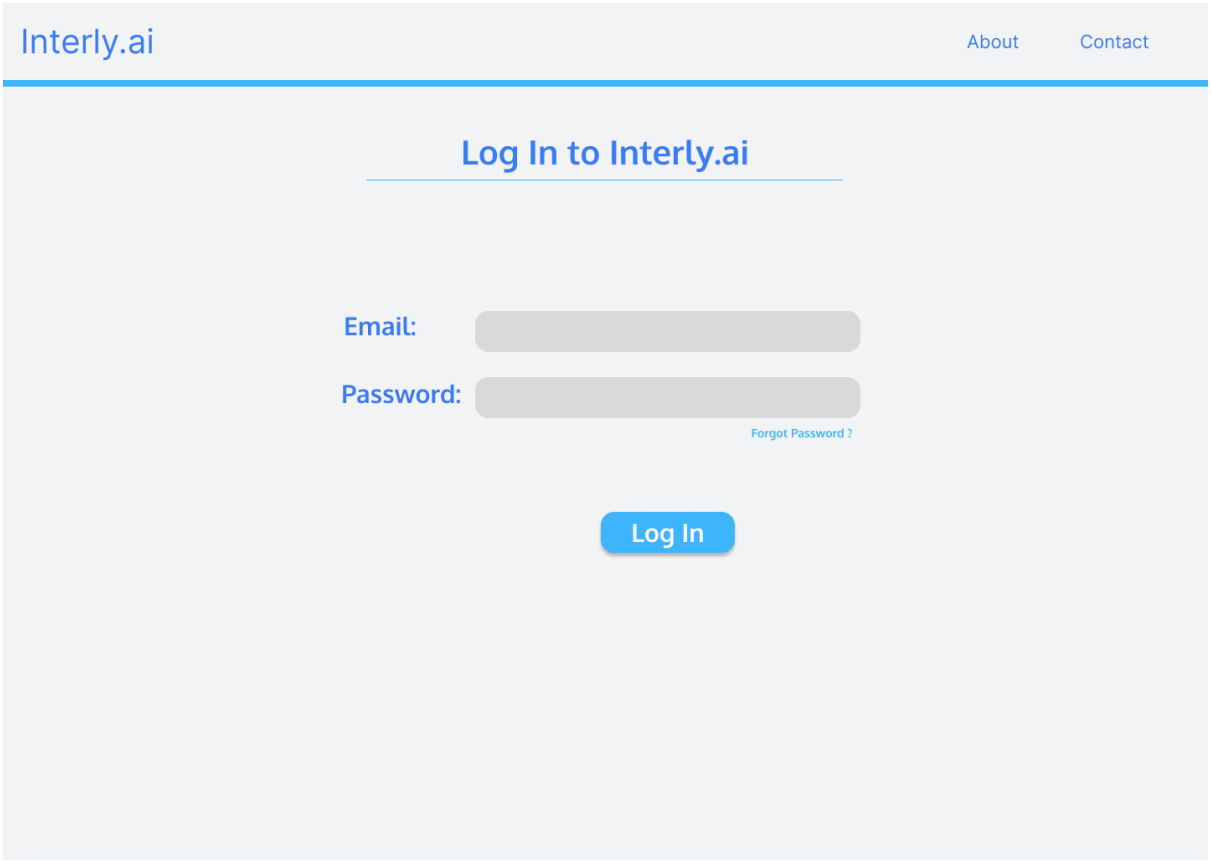
To create an interview, the user has to have either an HR account or HR Manager account. This scenario will show you how to create an interview with an HR account. However, creating an interview with an HR Manager account is also similar to this scenario.

Landing Page



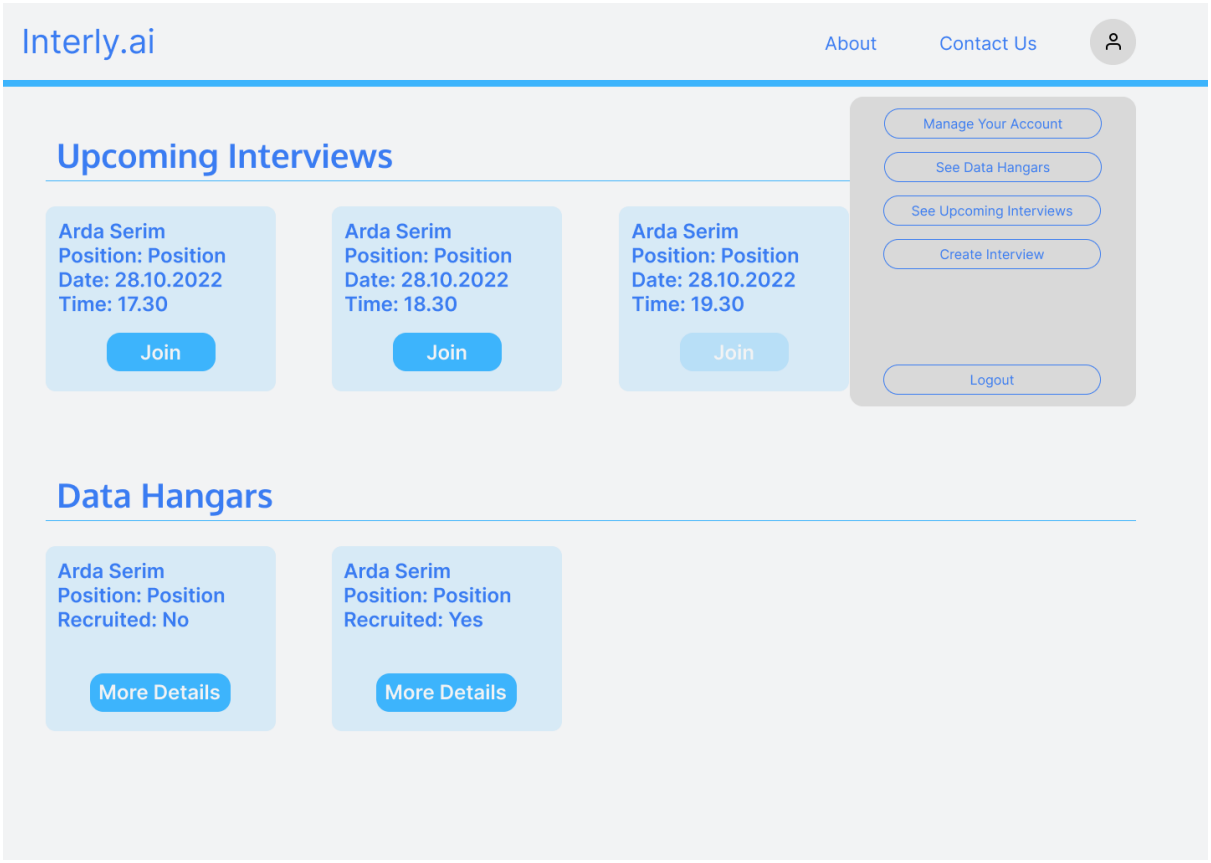
In the landing page, click the sign in button.

Login Page



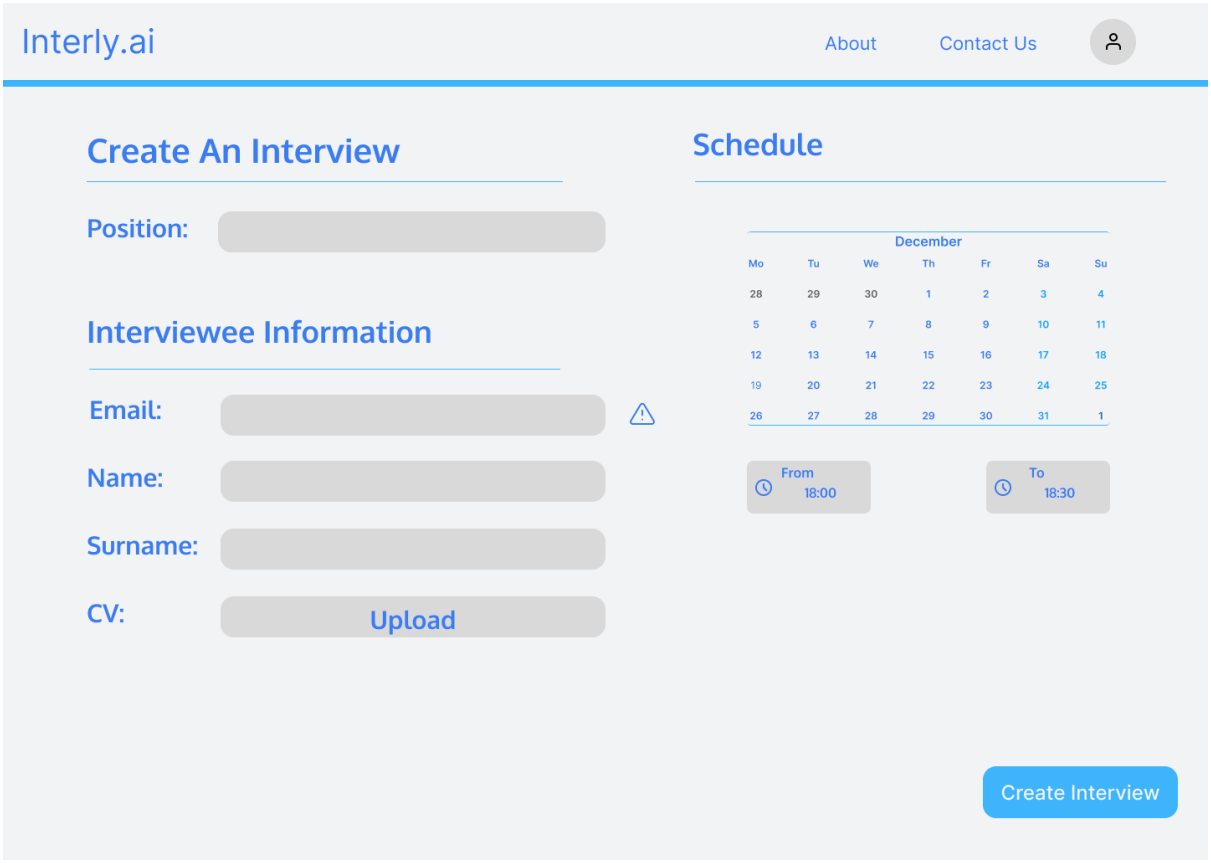
Then, sign in with your credentials.

HR Main Page:



You will be directed to the main page. In this page, click the button on the right top of the page. It will open a menu. In this menu click the create interview button.

Interview Creation

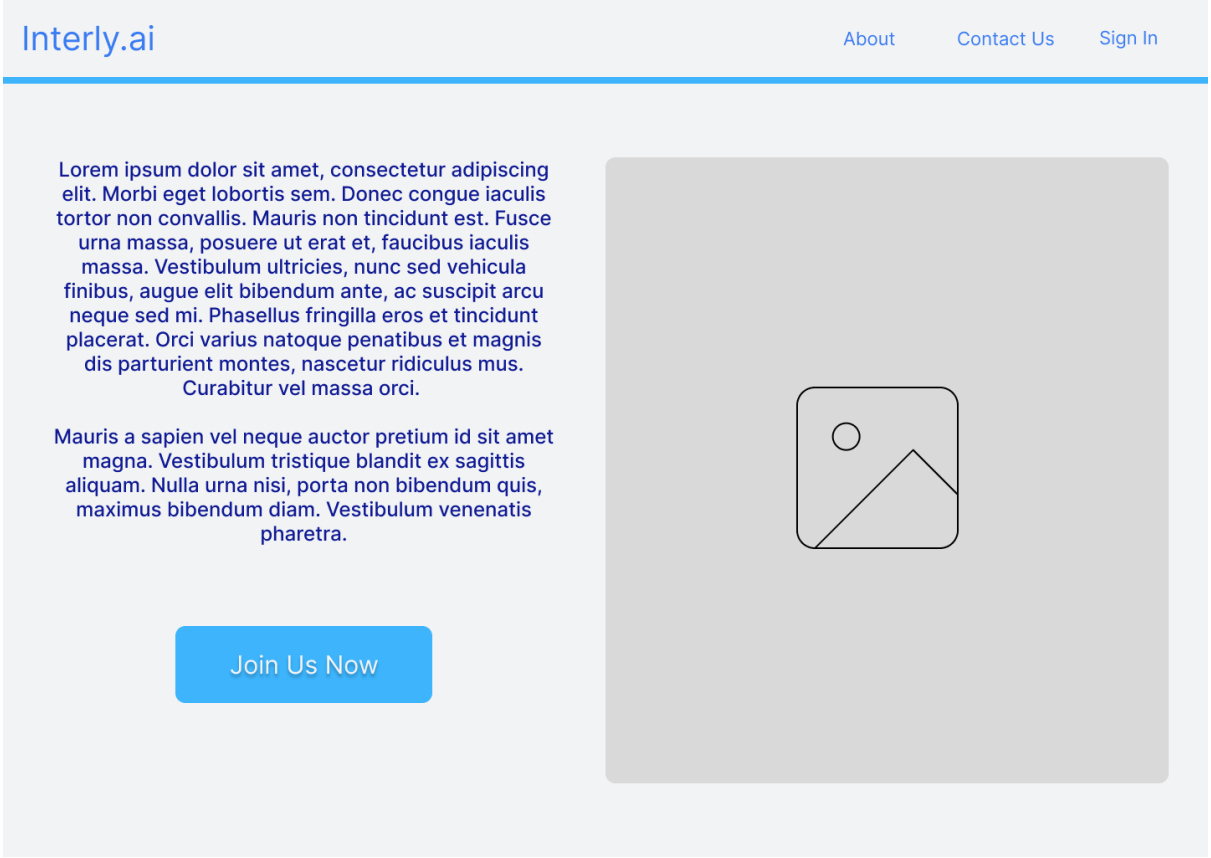


You will be redirected to the interview creation page. In this page, you can enter the position, interviewee's information (email, name, surname and CV). Then you need to choose a day and start and end time. After completing these steps, click the create interview button.

Join a meeting

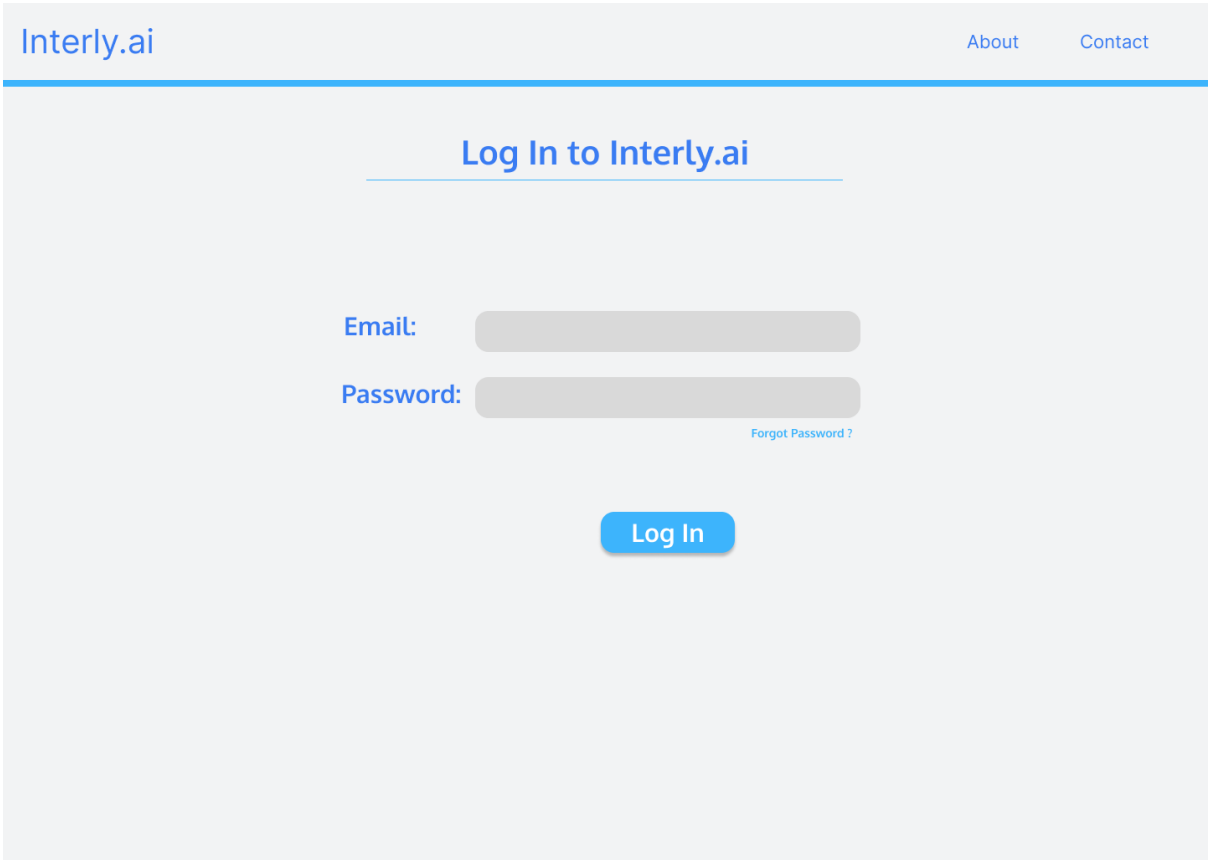
To join a meeting, the user has to have either an HR account or HR Manager account. This scenario will show you how to join a meeting with an HR account. However, joining a meeting with an HR Manager account is also similar to this scenario.

Landing Page



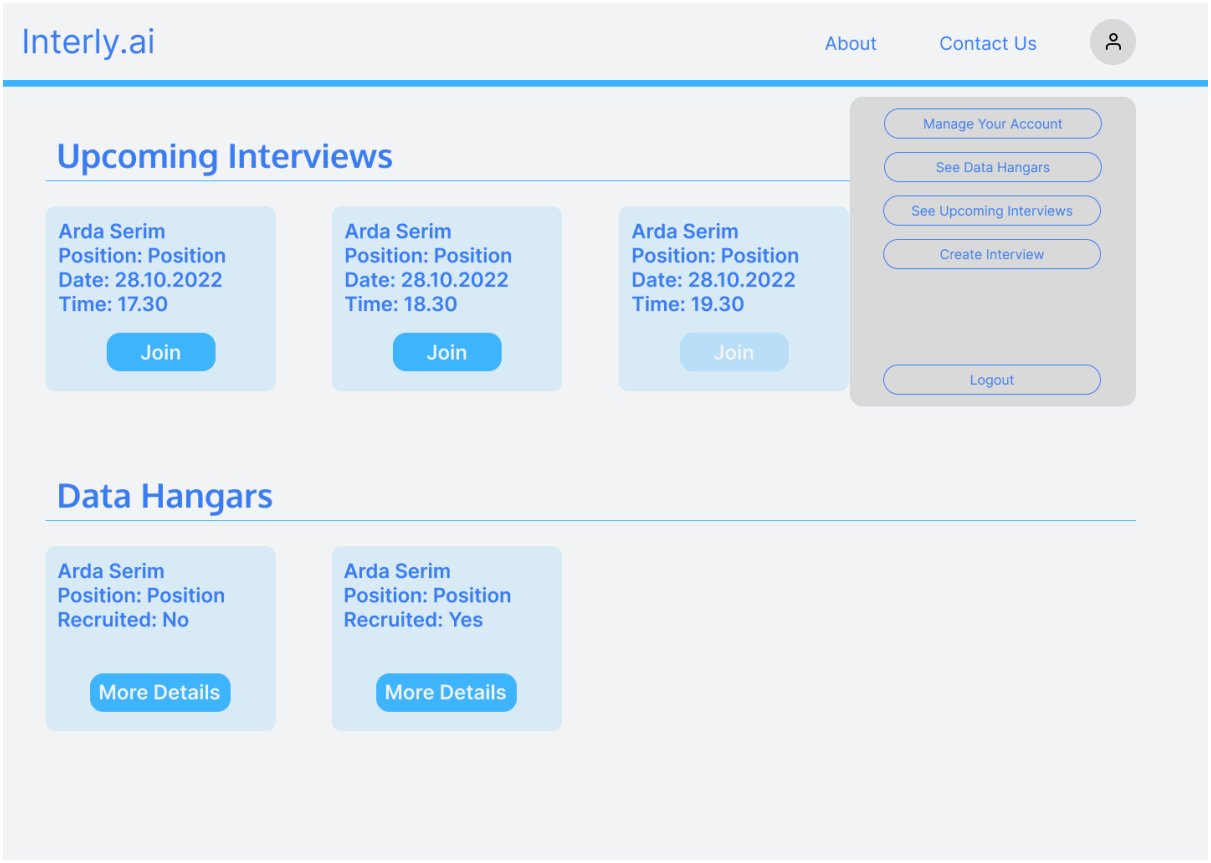
In the landing page, click the sign in button.

Login Page



Then, sign in with your credentials.

HR Main Page:

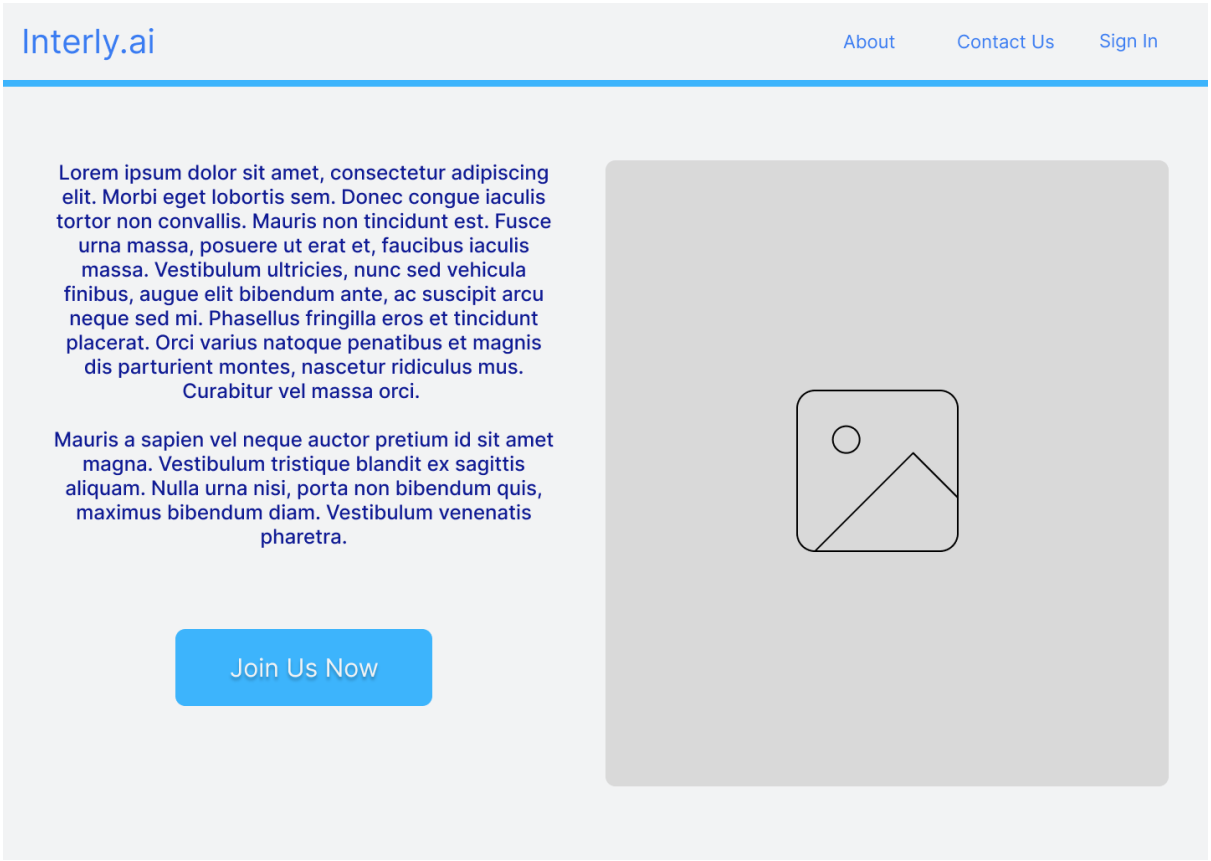


You will be directed to the main page. In this page, you can see the upcoming interviews. If the join button is not gray, you can click the button to join the meeting. If you want to see other meetings, you can use the wheel of the mouse to scroll the meetings.

Check Data Hangar

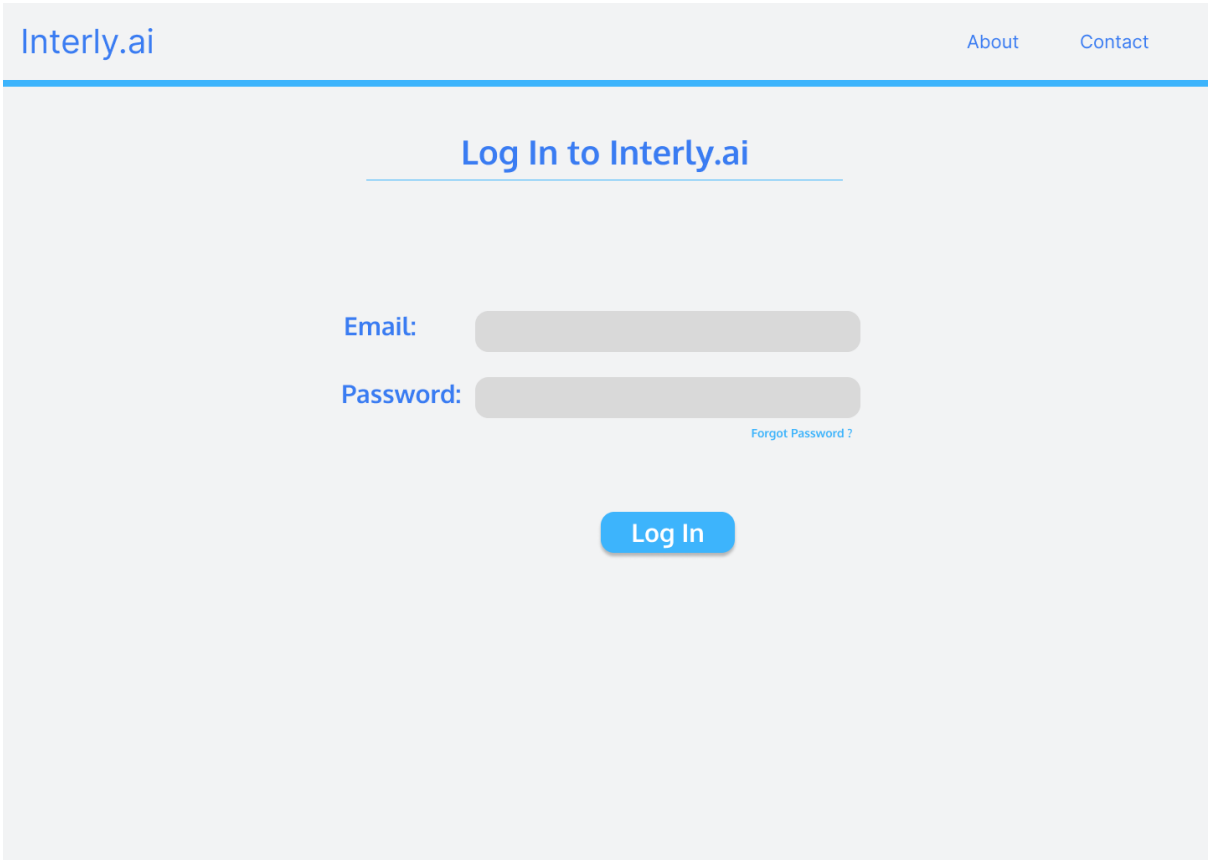
To join a meeting, the user has to have either an HR account or HR Manager account. This scenario will show you how to join a meeting with an HR account. However, joining a meeting with an HR Manager account is also similar to this scenario.

Landing Page



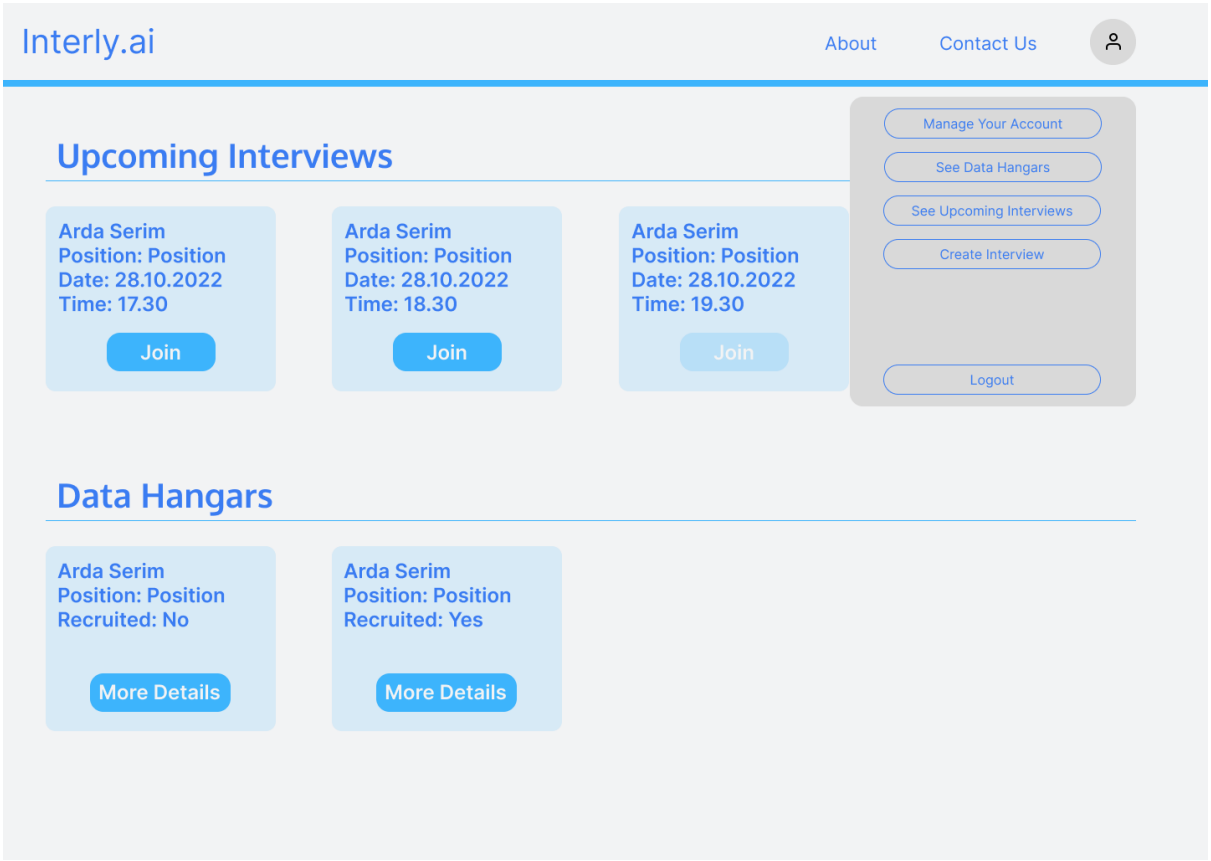
In the landing page, click the sign in button.

Login Page



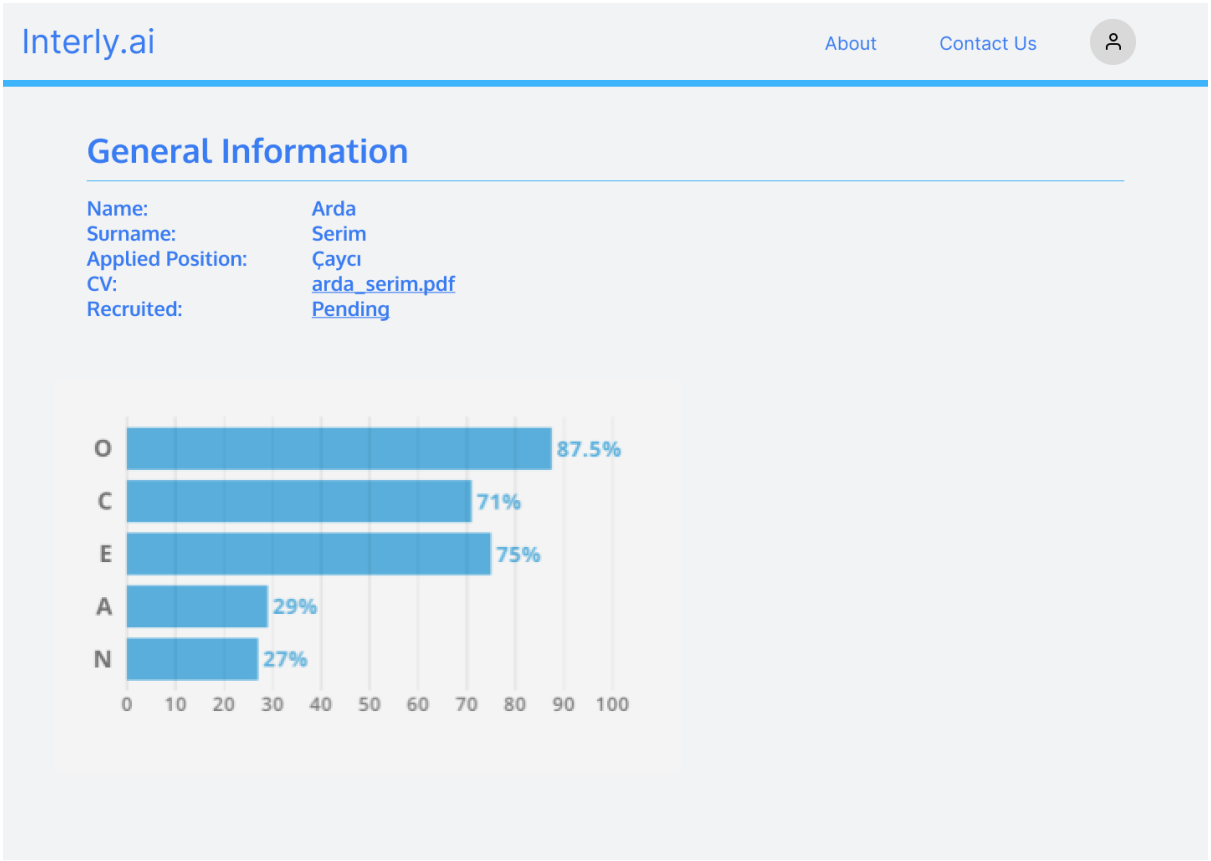
Then, sign in with your credentials.

HR Main Page:



You will be directed to the main page. In this page, you can see data hangars. It shows basic information such as interviewee name, position and whether the interviewee is recruited or not. When you click the more details button, you will be redirected to the data hangar page which shows details.

Data Hangar



HR can see all the details of the specific interview in its data hangar. It includes general information about the interviewee and the other information about the emotions.

Installation

When the interviewer creates an interview, an email will be sent to the interviewee. The email contains a download link of the necessary executable. The executable includes essential models for analysis. When the interviewee runs the executable, the interviewee is redirected to the meeting page.

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] “Introducing the Jitsi Meet React SDK”. *Jitsi*. [Online]. Available: https://jitsi.org/blog/introducing-the-jitsi-meet-react-sdk/ . [Accessed: 15-May-2023].

[4] S. Justin, “fer: Facial expression recognition from images,” *PyPI*. [Online]. Available: https://pypi.org/project/fer/ . [Accessed: 19-Nov-2022].

[5] O. Jonathan, “Face expression recognition dataset”. *Kaggle*. [Online]. Available: https://www.kaggle.com/datasets/jonathanoheix/face-expression-recognition-dataset?resource=download . [Accessed: 20-Mar-2023].

[6] “TruthSayer: A Remote Lie-detector,” Github. [Online]. Available: https://github.com/everythingishacked/Truthsayer . [Accessed: 17-May-2023].

[7] “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].