System Design Specification

For

PSYCHOMETRIC ANALYSIS WEB TOOL

Prepared by:

Apurva Bhargava(BE/25022/15), Pallavi Jain(BE/25001/15)

Birla Institute of Technology, Mesra, Jaipur Campus

12 March 2019

Table of Contents

1. INTRODUCTION

This document is designed to be a reference for any person wishing to implement or any person interested in the architecture of the software. This document describes the application's architecture and sub architecture, the associated interfaces, the database and the motivations behind choosing the design. Both high level and low level designs are included in this document. This document should be read by an individual with a technical background and has experience reading data flow diagrams, control flow diagrams, interface designs and development experience in object-oriented programming as well as sequential programming.

The document will provide developers an insight in meeting client's needs efficiently and effectively. It would demonstrate how the design will accomplish the functional and nonfunctional requirements captured in the SRS.

1.1 Document Description

It is the "how will we do it" part after we wrote the "what will we do" part, which is the SRS.

The System Architecture section is the main focus of this document. It provides an overview of the system's major components and architecture, as well as specifications on the interaction between the system and the user.

The Detailed System Design description of components section will also be covered in this document. It will describe lower-level classes, components, and functions, as well as the interaction between these internal components. It contains specific information about the expected input, output, classes, and functions. The interactions between the classes to meet the desired requirements are outlined in detailed figures (class diagram) at the end of the document. It supplies a snapshot of the intended system from multiple points of view.

Here is the outline of the proposed software design specifications.

- Introduction
- System Overview
- Design Considerations
    - Assumptions and Dependencies
    - General Constraints
    - Goals and Guidelines
    - Development Methods
- System Architecture
    - Client-side or Frontend Implementation
    - Server-side or backend Implementation
    - Face Expression and Speech Emotion Recognition Models
    - Sentiment Analysis Model
    - Adaptive Interview Chatbot
- Detailed System Design
    - Web Implementation

    - Real-time Emotional State Determination
    - Image Description-based Sentiment Analysis
    - Adaptive Interview Chatbot
- Use Case Diagrams
- Dataflow Diagrams
- Bibliography

1.2 System Overview

Psychometrics is the science of assessment of mental capacities and processes. Psychometric Analysis Web Tool powers judgment based on the assessment of emotional state of an individual by using webcam, microphone and keyboard input. It is a tool that can be used for assistance to existing interviewers and/or psychologists which aims at provisioning solutions to administer the same in the absence of a human interviewer and/or psychologist. It acts as the interface between the two interacting parties, one of which evaluates the other using the various modules provided by the software as mentioned further. It can be widely used by the companies for recruitment purposes. Also, a psychologist can use it for evaluation of emotional state. This is achieved by various modules that analyze the responses of the interviewee or patient in the form of text, speech, facial emotion, etc.

2. DESIGN CONSIDERATIONS

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

2.1 Assumptions and Dependencies

- For face and speech emotion classification, the seven classes are anger, disgust, fear, surprise, sadness, happiness and neutral.
- For text, the sentiment classes are positive, negative and neutral.
- Webcamera and microphone (inbuilt or otherwise) are available.
- The combined observations from different modules are enough to reach to a conclusion.
- There are no dependencies that the project has on external factors.

2.2 General Constraints

- **Operating Environment:** The tool can run on any browser compatible with ReactJS v14.9 on any operating system environment.
- **Performance Requirements:** The system shall be interactive and the delays involved shall be set to a minimum. So, in every action-response of the system, there are no

immediate delays. In case of submissions or saving sessions like saving tests, there is a delay of duration below 1 second.

- **Security Requirements:** Information transmission shall be securely transmitted to server/ database without any changes in information. Test administrators and users or test takers have different account types. This allows for different functionalities, restricts access to file systems and preservation of settings of a particular administrator or user.
- **Availability:** The tool runs on a webpage and is independent of network connections pertaining to the node system on which it is installed. It can be run on any compatible browser on any operating system environment.
- **Usability:** The web tool is easy to handle and navigates in the most expected way with no delays. The built is responsive. The APIs and server handle requests swiftly.
- **User Interfaces:** The administrator is expected to be familiar with the interface of the system. A simple GUI interface that has a separate window for separate tasks, gives an enhanced test experience to both the user and the responder.
- **Hardware Interfaces:** Reliable software device drivers usually handled by the OS distributors are required to run the webcamera, keyboard, mouse, and microphone for recording responses. All devices compatible with modern computing systems are supported.
- **Software Interfaces:** The Psychometric Analyzer has different types of software interfaces (this term is used in a very broad meaning) to external packages, depending how the interaction is realized:
    i. User Interface: The various functionalities of the application can be accessed by the means of simple buttons, tabs and forms.
    ii. Message Interface: Since automation is the goal, methods have been implemented to run the various functionalities and communicate among different objects (instances of different classes) using Application Programming Interfaces (APIs).
    iii. Database Interface: The test administrator can create and modify test and add, remove or modify images and questions-answers in the file system.

2.3 Goals and Guidelines

- Creating an online tool or product that assists a human interviewer and/or psychologist and automates a large part of psychometric assessment.
- Classification into emotions/ sentiment and using some heuristic or guidelines to understand their significance.
- Combination of observations from different modules to reach to a conclusion.
- The efficiency of training of the model used for analysis depends on the CPU/GPU used and hence powerful ones are recommended.
- A properly working web camera and microphone are must for this to function appropriately.

2.4 Development Methods

Function-oriented design is considered for the design, wherein, each feature or utility of the tool is implemented as a function. The architectural model is client-server where in backend utilities are run by making calls using APIs. Responsive web design is used which an approach to web design that makes web pages render well on a variety of devices and window or screen sizes.

3. SYSTEM ARCHITECTURE

This section provides a high-level overview of how the functionality and responsibilities of the system are partitioned and then assigned to subsystems or components. The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.

At the top-most level, the description of the major responsibilities that the software undertakes and how the higher-level components collaborate with each other in order to achieve the required results are given.

The overall website architecture is based on the Client–server model- a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients.

3.1. Website Frontend

The client-side implementation uses ReactJS and NodeJS frameworks. There are four page types- Home page, Register and Sign-in page, Logged-in page and Dashboard page. All the functions including the clicks, registration, signing up, navigation, inputs on the website are actions that are of importance in determining the behaviour, responses and updates made to the database and this is how they move forward in their journey. Three-tier client server architecture is used because data is stored in database (File System) can be accessed and modified by the admin, the client with the help of the GUI between them. Frontend output is the response to any event or action performed on the website that can be seen in the form of routing to different pages, the change of questions or images on next and previous buttons, starting, submitting and moving through the phases of the test, etc.

3.2. Website Backend

The server-side implementation uses NodeJS and Django (Python) frameworks. The file system interaction is handled by the NodeJS, for example, saving the images in folders, question-answers in .json files in separate folders for each user for each test. The processing of inputs and their responses in terms of score is generated by the python server and returned to the frontend. All the inputs to the NodeJS server result in either publishing the data to files and folders or fetching it from there. The question-answers are saved as .json file, images are saved in the folder inside Admin→Test. The responses of the user in different phases are also saved in the .json files. These responses are sent to python server for evaluation and determination of score by using ML models or Chatbot system.

3.3. Facial Expression and Speech Emotion Recognition Model

The Facial Expression Recognition Model is a convolutional neural network model with a sequence of convolution layers, followed by a max-pooling layer, and a dense layer with nodes = 1024 and dropout rate = 0.5 succeeding all the convolution and pooling layers. Lastly, a dense layer for 7 outputs with softmax activation follows. This model is trained using the FER2013 dataset and saved for use. The face detection component is implemented using Haar-feature based cascade classifier using cv2 (OpenCV). A rectangular frame is drawn around the detected part in real-time video feed from webcam. Detected sections of images with faces, captured periodically, are resized to 48x48 pixels, improved by increasing brightness and converted to grayscale. These grayscales are fed to the trained neural network model for classification of facial expressions into disgusted, angry, sad, happy, surprised, fearful and neutral. The results of analysis are displayed on the GUI. Libraries for FER model: keras, numpy, sklearn, cv2.

This component records audio from microphone periodically and feeds it to the trained stacked LSTM model for classification of speech emotions into disgust, anger, sadness, happiness, surprise, fear and neutral. The LSTM model is trained using the RAVDESS and SAVEE dataset. Libraries used: scipy for audio extraction and keras for training.

3.4. Sentiment Analysis Model

This analysis requires the responder to input a description of a displayed image in about 100 words or some other specified limit. This text is then pre-processed by removing punctuations, extending contractions, POStagging for selecting sentiment descriptors, lemmatizing and stemming. From a similar pre-processing, followed by CountVectorizer and employing a self-designed variation of TF-IDF, a lexicon was generated. For the response as well, the lexicon is used for extracting numerical features. These features are fed to the trained sentiment analysis model (datasets: Stanford Movie Reviews, Amazon Reviews, Twitter Airline Sentiment) for estimation of the positive, negative and neutral perception percentages of an individual in response to the respective images. The results are displayed on the GUI window. Libraries used: nltk, numpy, pandas, csv for pre-processing, and keras for model.

3.5. Adaptive Interview Chatbot

Chatbot is trained using .yml files containing conversations, and later consults sqlite database generated from the .yml files. The administrator’s questions, answers, difficulties, and maximum scores from file system are used to generate a .json file. For each question, multiple similar answers are generated using the expected answer. Question Answering or Q/A system parses the .json file to select questions and match responses. A confidence value (using context) is used to decide which system responds. Questions are chosen based on difficulty and user’s performance. Libraries used: chatterbot, nltk, thesaurus.

4. DETAILED SYSTEM DESIGN

Most components described in the System Architecture section will require a more detailed discussion. Other lower-level components and subcomponents may need to be described as well. Each subsection of this section will refer to or contain a detailed description of a system software component.

4.1. Web Implementation

LANDING PAGE OR HOME PAGE: It is the first page which lands the visitor to discover the features and concept of the tool. From Register page, a user/ admin account can be created, saved and File Storage System initialized. From the Sign-in page, the user/ admin can sign in with credentials verification and access the database and tests.

LOGGED-IN PAGE (FOR ADMIN): The test administrator can create a test (add test name and test description, questions, answers and images), modify the test (modify test description, questions, answers and images), specify the test takers and see their responses and results.

DASHBOARD PAGE (FOR USER): The user can select a test (using test name and test admin), give the test and see the results.

TEST PHASES:

Phase 1: For several questions/ prompts, the test taker's images will be periodically captured from webcamera and the audio recorded for emotion recognition using Facial Expression Recognition and Speech Emotion Recognition models.

Phase 2: For all images, the user has to write a description in the given box. The sentiment classification model analyses the description and assign percentages to the sentiment classes.

Phase 3: Bot starts the conversation and asks questions. User responds in a message box below the conventional scrollable chat window. Response is submitted by pressing Enter key.

Each phase has its individual results and analyses section.

4.2. Real-time Emotional State Determination

MODELS: FER model (CNN) trained using FER-2013 to 63% accuracy and SER model (LSTM) trained using RAVDESS to 83% accuracy. Emotion classes: angry, fearful, disgusted, happy, surprised, sad and neutral.

FRONTEND: React modules: react-webcam and react-mic for video and audio feed.

BACKEND: The feed from frontend is sent to Django API which does the processing and returns the percentages of the 7 emotion classes.

4.3. Image Description-based Sentiment Analysis

MODEL: Sentiment classification model (ANN) trained using Amazon product reviews, Stanford movie reviews and Twitter Airline tweets to 93% accuracy. Sentiment classes: positive, neutral and negative.

FRONTEND: The images from the database (Node Server) are fetched from the file system and displayed. A text area where the description can be written.

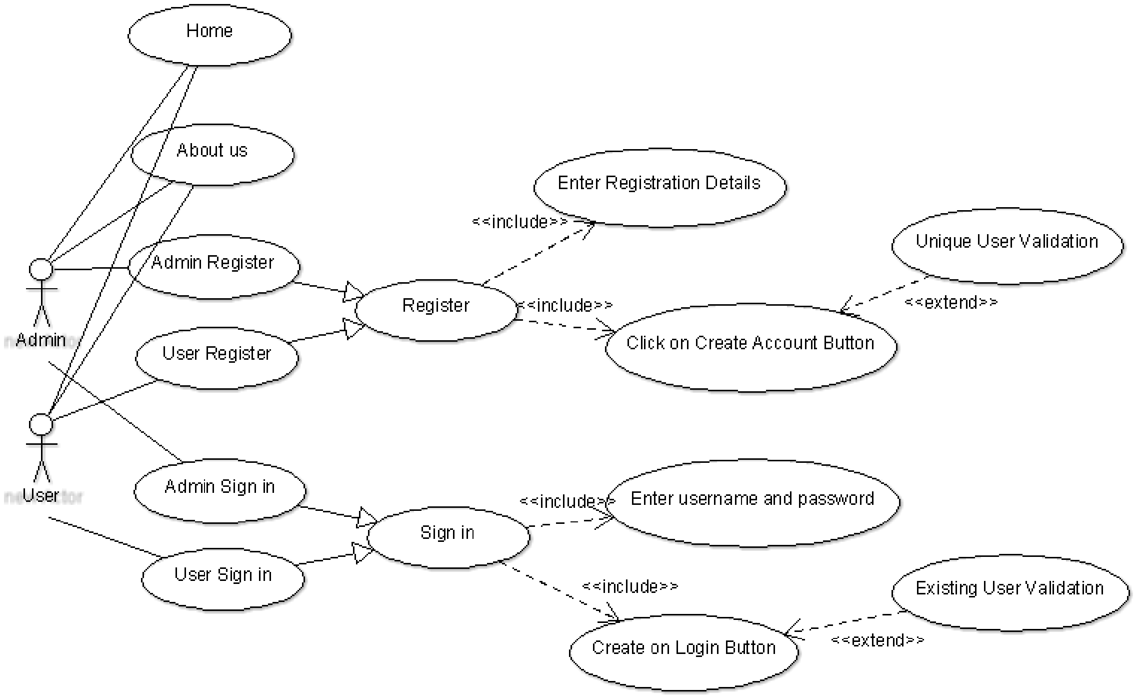
BACKEND: The input from frontend is sent to the API (Django Server) for processing and it return the sentiment scores of three classes.

4.4. Adaptive Interview Chatbot

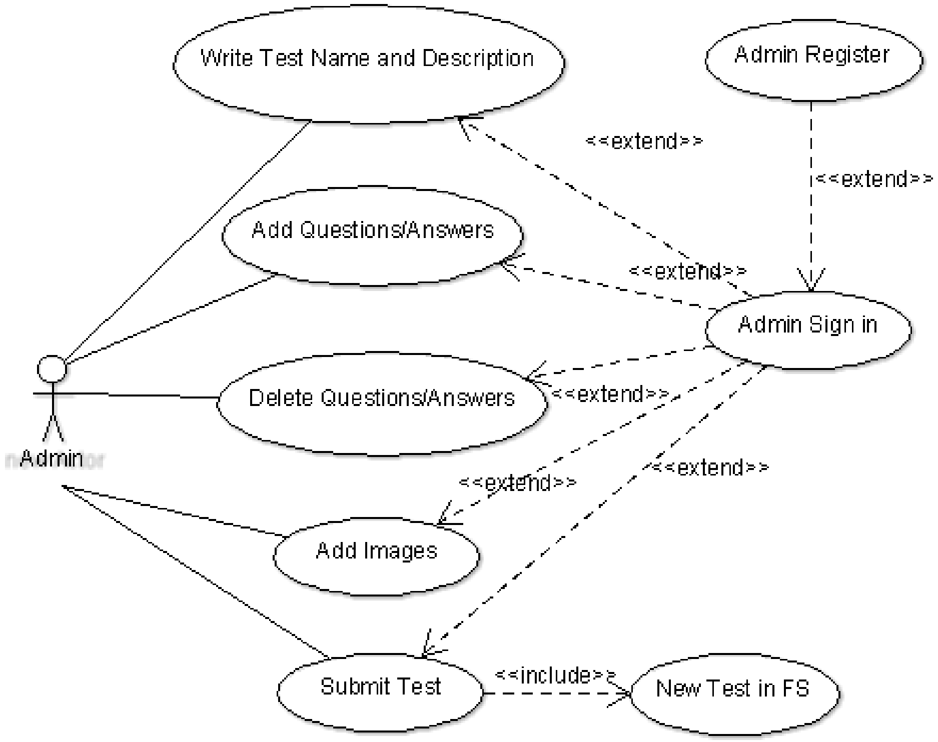
MODEL: Chatbot is trained using .yml files containing conversations under conversations tag and a categories tag. A sqlite database is generated from the .yml files. The chatbot matches input and searches responses from the sqlite database. The administrator’s questions, answers, difficulties, and maximum scores from file system are used to generate a .json file. For each question, multiple similar answers with different synonyms replacing actual words are generated using the expected answer. Question Answering or Q/A system parses the .json file to select questions and match responses. Both the chatbot and Q/A system run in parallel and process user input. A confidence value (using context) is used to decide which system responds. Questions are chosen based on difficulty and user’s performance on previous question.
FRONTEND: A conventional scrollable chat window
BACKEND: User’s response is sent to server and bot’s response is fetched and displayed.

5. USE CASE DIAGRAMS

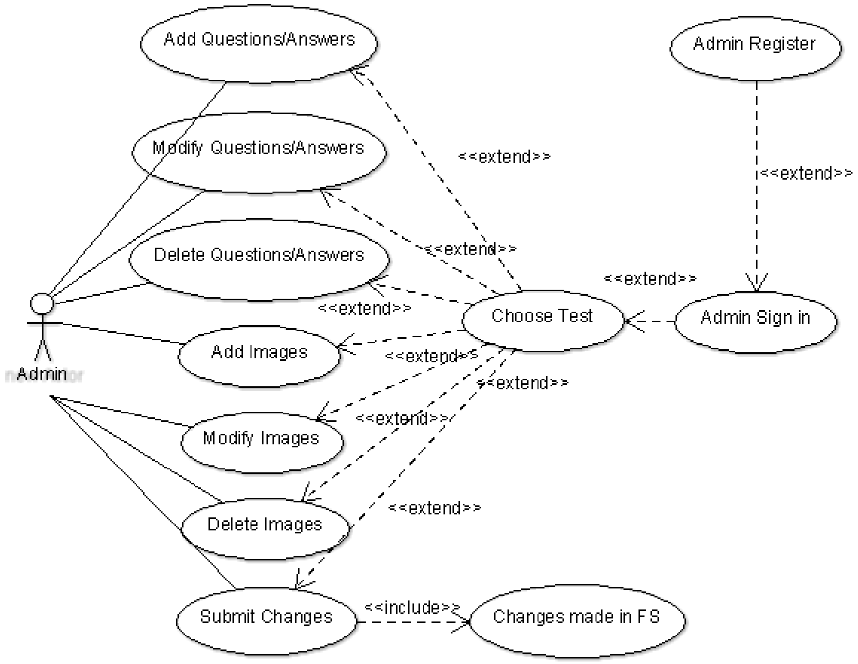
5.1. Functionality 1 (Register and Sign in)



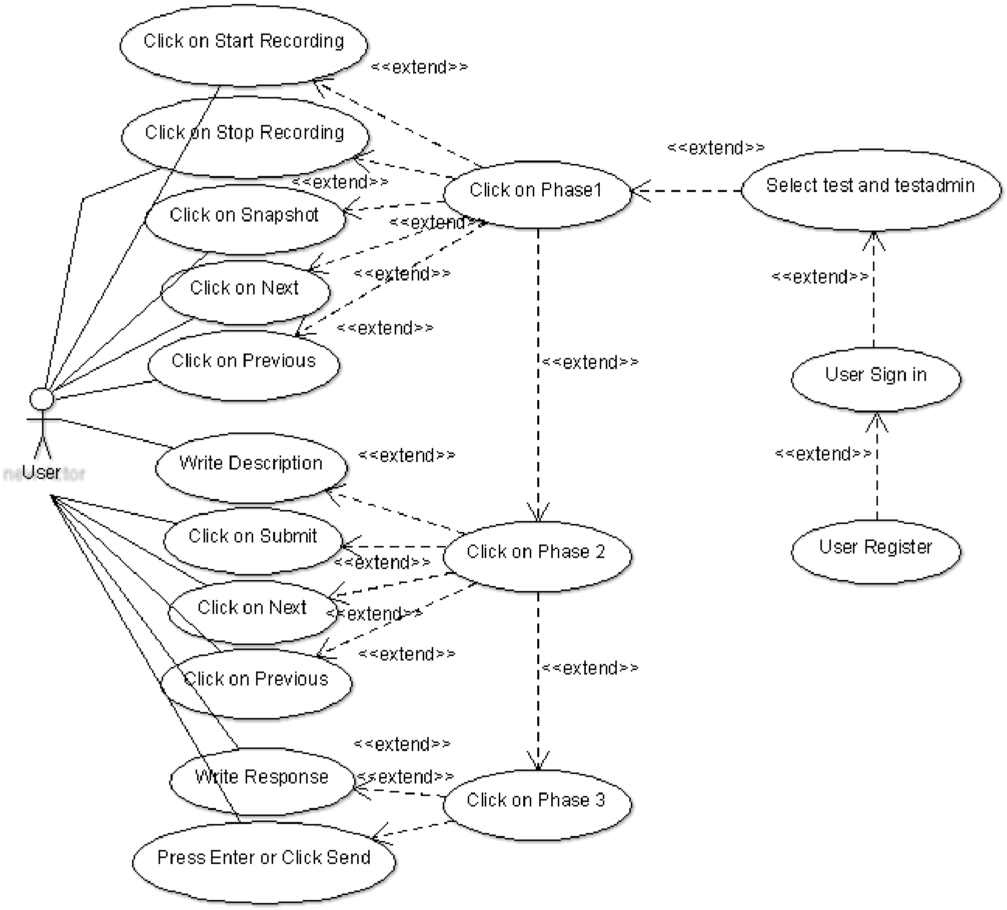
5.2. *Functionality 2 (Create Test)*



5.3. *Functionality 3 (Modify Test)*

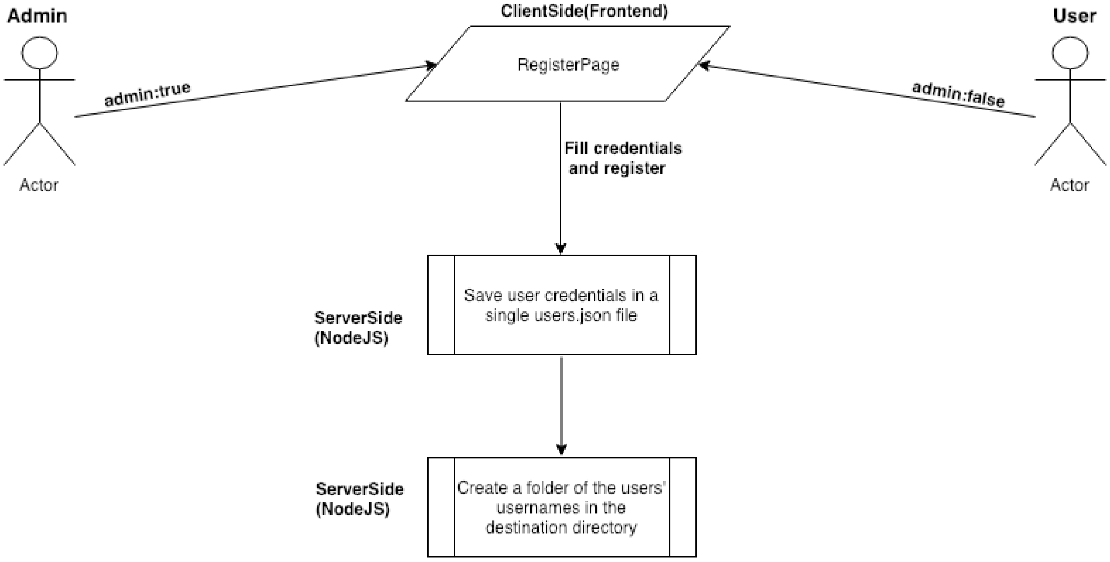


5.4. Functionality 4 (Take Test)

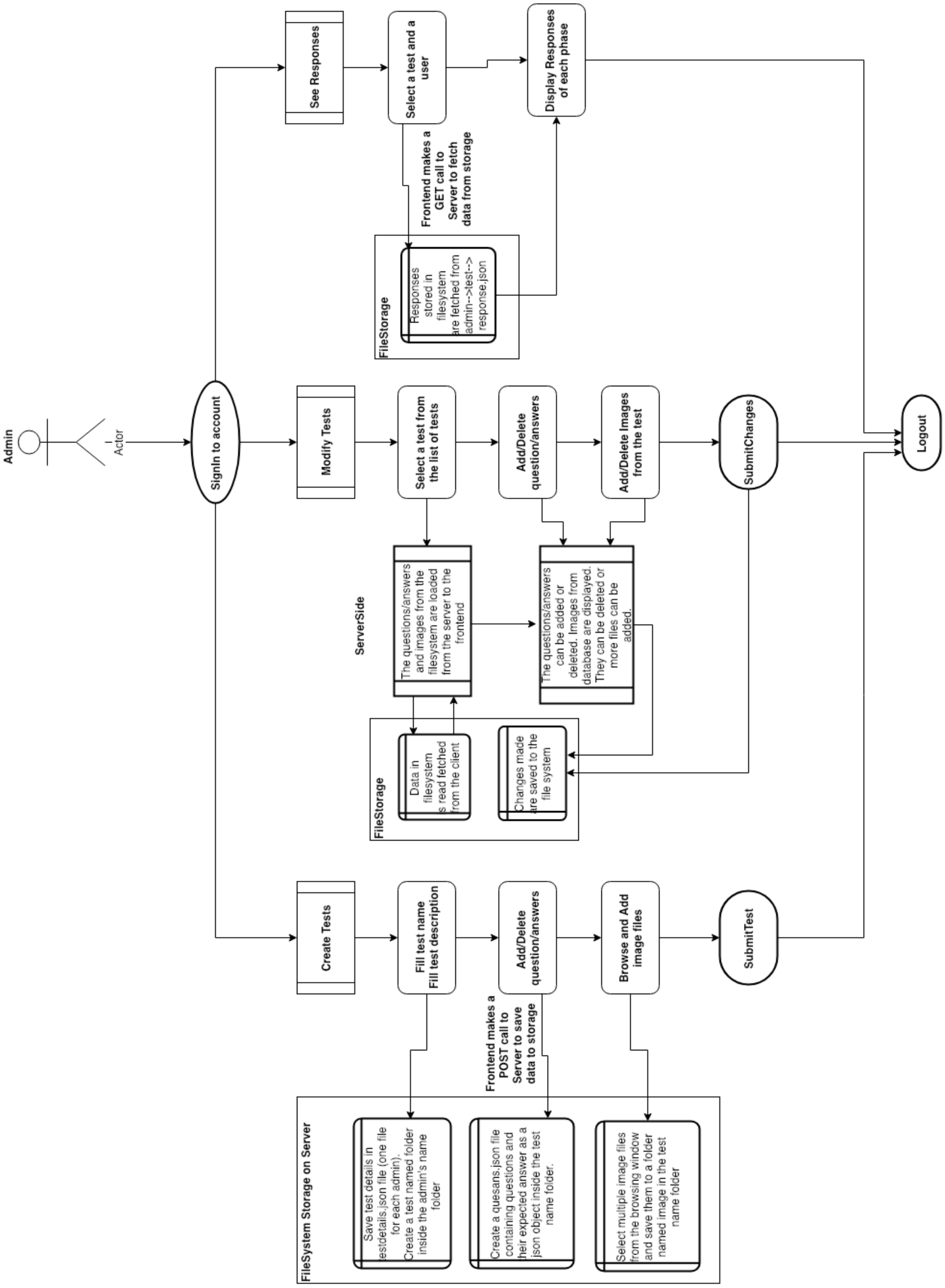


6. DATAFLOW DIAGRAMS

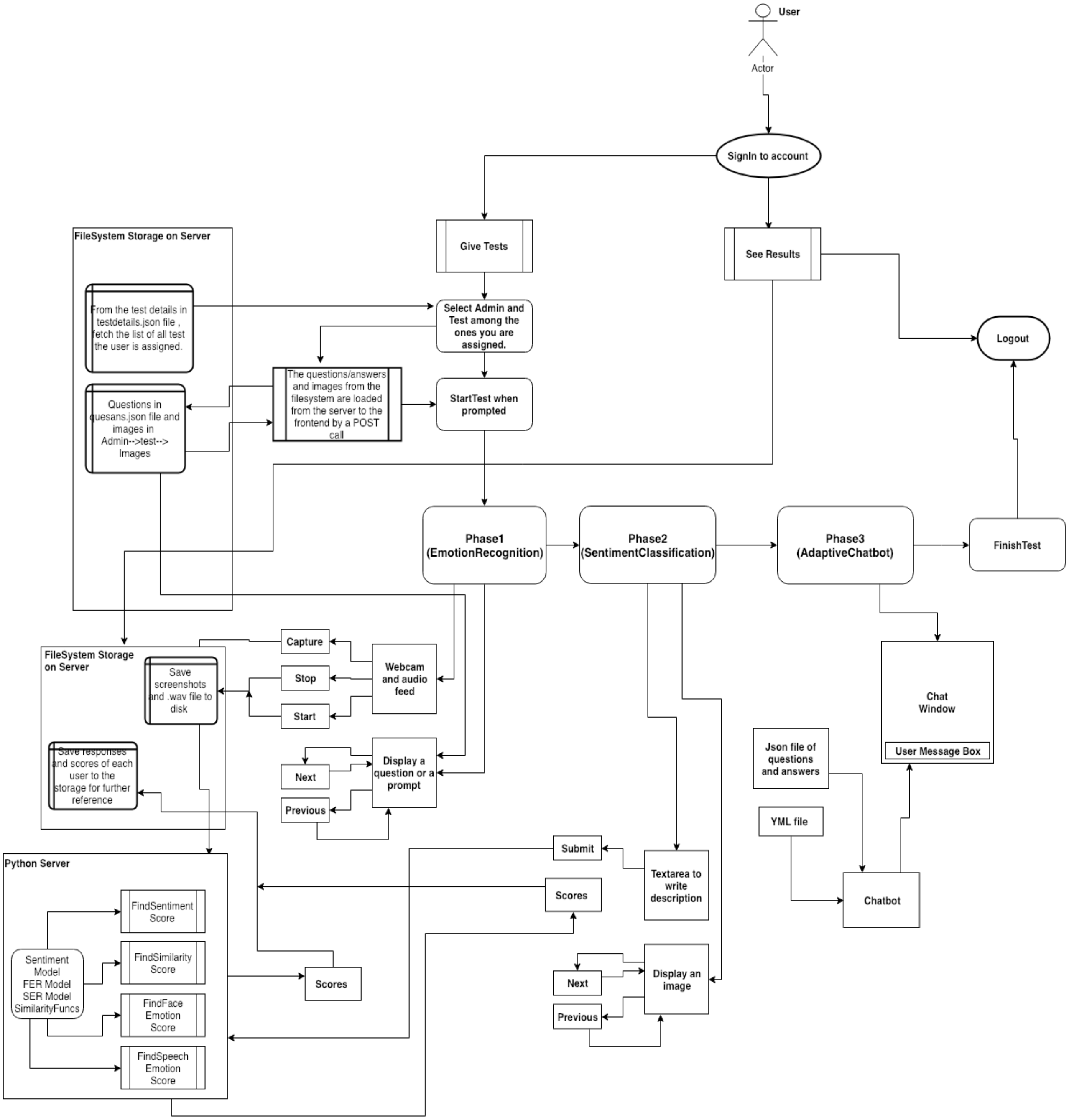
6.1. Dataflow for Registration


Admin
Actor
admin:true
ClientSide(Frontend)
RegisterPage
User
Actor
admin:false
Fill credentials and register
ServerSide (NodeJS)
Save user credentials in a single users.json file
ServerSide (NodeJS)
Create a folder of the users' usernames in the destination directory

6.2. Dataflow for Administrator Interaction


Admin
Actor
SignIn to account
See Responses
Select a test and a user
Display Responses of each phase
Frontend makes a GET call to Server to fetch data from storage
FileStorage
Responses stored in filesystem are fetched from admin-->test--> response.json
Modify Tests
Select a test from the list of tests
Add/Delete question/answers
Add/Delete Images from the test
SubmitChanges
Logout
ServerSide
The questions/answers and images from the filesystem are loaded from the server to the frontend
The questions/answers can be added or deleted. Images from database are displayed. They can be deleted or more files can be added.
FileStorage
Data in filesystem s read fetched from the client
Changes made are saved to the file system
Create Tests
Fill test name Fill test description
Add/Delete question/answers
Browse and Add image files
SubmitTest
Frontend makes a POST call to Server to save data to storage
FileSystem Storage on Server
Save test details in estdetails.json file (one file for each admin). Create a test named folder inside the admin's name folder
Create a quesans.json file containing questions and heir expected answer as a json object inside the test name folder.
Select multiple image files from the browsing window and save them to a folder named image in the test name folder

6.3. Dataflow for User Interaction


User
Actor
SignIn to account
Give Tests
See Results
Logout
FileSystem Storage on Server
From the test details in testdetails.json file , fetch the list of all test the user is assigned.
Select Admin and Test among the ones you are assigned.
Questions in quesans.json file and images in Admin-->test--> Images
The questions/answers and images from the filesystem are loaded from the server to the frontend by a POST call
StartTest when prompted
Phase1 (EmotionRecognition)
Phase2 (SentimentClassification)
Phase3 (AdaptiveChatbot)
FinishTest
FileSystem Storage on Server
Save screenshots and .wav file to disk
Save responses and scores of each user to the storage for further reference
Capture
Stop
Start
Webcam and audio feed
Next
Previous
Display a question or a prompt
Chat Window
User Message Box
Json file of questions and answers
YML file
Chatbot
Submit
Scores
Textarea to write description
Python Server
FindSentiment Score
FindSimilarity Score
FindFace Emotion Score
FindSpeech Emotion Score
Sentiment Model FER Model SER Model SimilarityFuncs
Scores
Next
Previous
Display an image

7. REFERENCES

[1] National Council on Measurement in Education- http://www.ncme.org/ncme/NCME/ Resource_Center/Glossary/NCME/Resource_Center/Glossary1.aspx?hkey=4bb87415-44dc-4088-9ed9-e8515326a061#anchorPArchived 2017-07-22 at the Wayback Machine.

[2] Bishop, C.M. (2006), Pattern Recognition and Machine Learning, Springer, ISBN 978-0-387-31073-2 Zeiler, M.D. & Fergus, R.(2013). Visualizing and understanding convolutional networks. European Conference on Computer Vision(ECCV). 8689. 818-833.

[3] Jaswal, Deepika & Vishvanathan, Sowmya & Kp, Soman. (2014). Image Classification Using Convolutional Neural Networks. International Journal of Scientific and Engineering Research. 5. 1661-1668. 10.14299/ijser.2014.06.002.

[4] Zhang, Ting. (2018). Facial Expression Recognition Based on Deep Learning: A Survey. 345-352. 10.1007/978-3-319-69096-4_48.

[5] Schmidhuber, Juergen. (2014). Deep Learning in Neural Networks: An Overview. Neural Networks. 61. 10.1016/j.neunet.2014.09.003.

[6] Chernykh, Vladimir & Sterling, Grigoriy & Prihodko, Pavel. (2017). Emotion Recognition From Speech With Recurrent Neural Networks.

[7] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

[8] Cen, Ling & Dong, Minghui & Li, Haizhou & Chan, Paul. (2010). Machine Learning Methods in the Application of Speech Emotion Recognition. 10.5772/8613.

[9] Desai, Mitali & Mehta, Mayuri. (2016). Techniques for sentiment analysis of Twitter data: A comprehensive survey. 149-154. 10.1109/CCAA.2016.7813707.

[10] Das, Bijoyan & Chakraborty, Sarit. (2018). An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation.

[11] Yun-tao, Zhang & Ling, Gong & Yong-cheng, Wang. (2005). An improved TF-IDF approach for text classification. Journal of Zhejiang University - Science A: Applied Physics & Engineering. 6. 49-55. 10.1007/BF02842477.

[12] Khuat, Tung & Duc Hung, Nguyen & Thi My Hanh, Le. (2015). A Comparison of Algorithms used to measure the Similarity between two documents. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET). 4. 1117-1121.

[13] Shoaib, Muhammad & Daud, Ali & Khiyal, Malik. (2014). An improved Similarity Measure for Text Doocuments.. Journal of Basic and Applied Scientific Research. 4. 215-223. 10.13140/2.1.4814.4006.

[14] Computer Engineering and Applications Vol. 5, No. 1, February 2016. Comparison Jaccard similarity, Cosine Similarity and Combined Both of the Data Clustering With Shared Nearest Neighbor Method Lisna Zahrotun.

[15] https://keras.io/optimizers/

[16] https://keras.io/getting-started/sequential-model-guide/

[17] https://www.sqlite.org/docs.html

[18] https://people.csail.mit.edu/hubert/pyaudio/docs/

[19] https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recog nition -challenge/data

[20] https://smartlaboratory.org/ravdess

[21] http://jmcauley.ucsd.edu/data/amazon/

[22] https://nlp.stanford.edu/sentiment/index.html

[23] https://www.kaggle.com/crowdflower/twitter-airline-sentiment

[24] https://chatterbot.readthedocs.io/en/stable/

[25] https://pypi.org/project/thesaurus/

[26] P. Kataria, K. Rode, A. Jain, P. Dwivedi, S. Bhingarkar. User Adaptive Chatbot for Mitigating Depression. International Journal of Pure and Applied Mathematics. Volume 118 No. 16 2018, 349-361.

[27] https://reactjs.org/docs/getting-started.html

[28] https://nodejs.org/docs/latest-v8.x/api/

[29] https://docs.djangoproject.com/en/2.1/

[30] https://www.npmjs.com/package/react-webcam

[31] https://www.npmjs.com/package/react-mic