

AUDIO BID FINAL REPORT

PROJECT TEAM

Ajinkya Bharat Malhotra
Swatthi Vijay Sanker
Bhavya Rushin Shah
Abhiraj Singh Rathore
Neel Akash Murugesan Sathya Bama
Vasu Naman Verma

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
TEXAS A&M UNIVERSITY, COLLEGE STATION
DECEMBER 2022

CSCE 606 - Software Engineering
Instructor: Dr. Philip Ritchey
Fall 2022

Important Links:

Website	https://audiobid.herokuapp.com/
Pivotal Tracker	https://www.pivotaltracker.com/n/projects/2604793
GitHub Repo	https://github.com/VasuNVerma/Audio-Bid
Slack	https://cloud-6.slack.com
Poster and Demo Video	https://youtu.be/32RrSIOP1So

Summary:

The project, Audio Bid is an online audio transcription job portal that helps workers to find jobs they can do and get paid for. The important stakeholders for this application include our client, Dr. John Cannavo, the creators/customers who will use this website to post job openings, and the workers/users who will bid on the available jobs. Our customers are the users and manager of the Mays Innovation Research Centre. The customer needs of this application were to be able to create jobs online, worker-bid on the available jobs where-in they can profit, review the completed jobs, rate the workers based on the completed jobs, and then send out payments to the workers.

We have created a web application that tries to meet the customer's needs. There will be two types of users in Audio Bid: workers and creators. There are separate logins for both these users. The portal allows the creation of jobs by the creator followed by a claim feature for the workers to start working on them. Once their work is submitted, the creator of the job can review the job by rating it and adding comments on the submitted task and accordingly pay the worker. The jobs are priced dynamically, which means that after every subsequent 24 hours, the price of the job increases to increase its demand. The portal provides a filter functionality to sort and extract jobs as required. We also have a profile section for each creator as well as worker, wherein all the user information is visible, along with their rating.

ITERATIONS

For each iteration, we will review the main user stories that were worked on, the details of the meeting held with the client, the team roles assigned, the lo-fi UI mock-ups created and the corresponding screen shots from the final website.

Iteration 0 [PO : Vasu Naman Verma SM : Abhiraj Singh Rathore]

Initially, our team Cloud 6 was assigned a legacy project “Reserve It”, which we got up and running and had set up the project in all our local devices to start working on as well. But during our second meeting with the client, we were asked to drop the project as there wasn’t sufficient documentation to continue working on it and we could not catch hold of anyone from the previous team.

Iteration 1 [PO : Bhavya Rushin Shah SM : Ajinkya Bharat Malhotra]

Iteration1 marked the start of our new project - “Audio Bid ”. We mainly focused on getting a broad picture of the requirements of our new client in our first client meeting held on 30th September 2022. The main aim of this iteration was to get our local development environments working along with the completion of certain basic user stories that would enable us to deploy a basic version of the application on Heroku.

Client Meeting Summary [09/30/2022]

The client gave us a broad picture of what he expects “Audio Bid” to look like. He used the “Job market” application as an example to illustrate the functionality.

Demo of the previous iteration	Current Iteration requirements
<ul style="list-style-type: none">No demo for this iteration as our project was changed from iteration 0 and this was the first meeting with our new client	<ul style="list-style-type: none">Login and Signup functionality for 2 different types of users - creator and worker.A basic homepage for each of the user types.Application deployed to Heroku for him to be able to demo it at the next meeting.

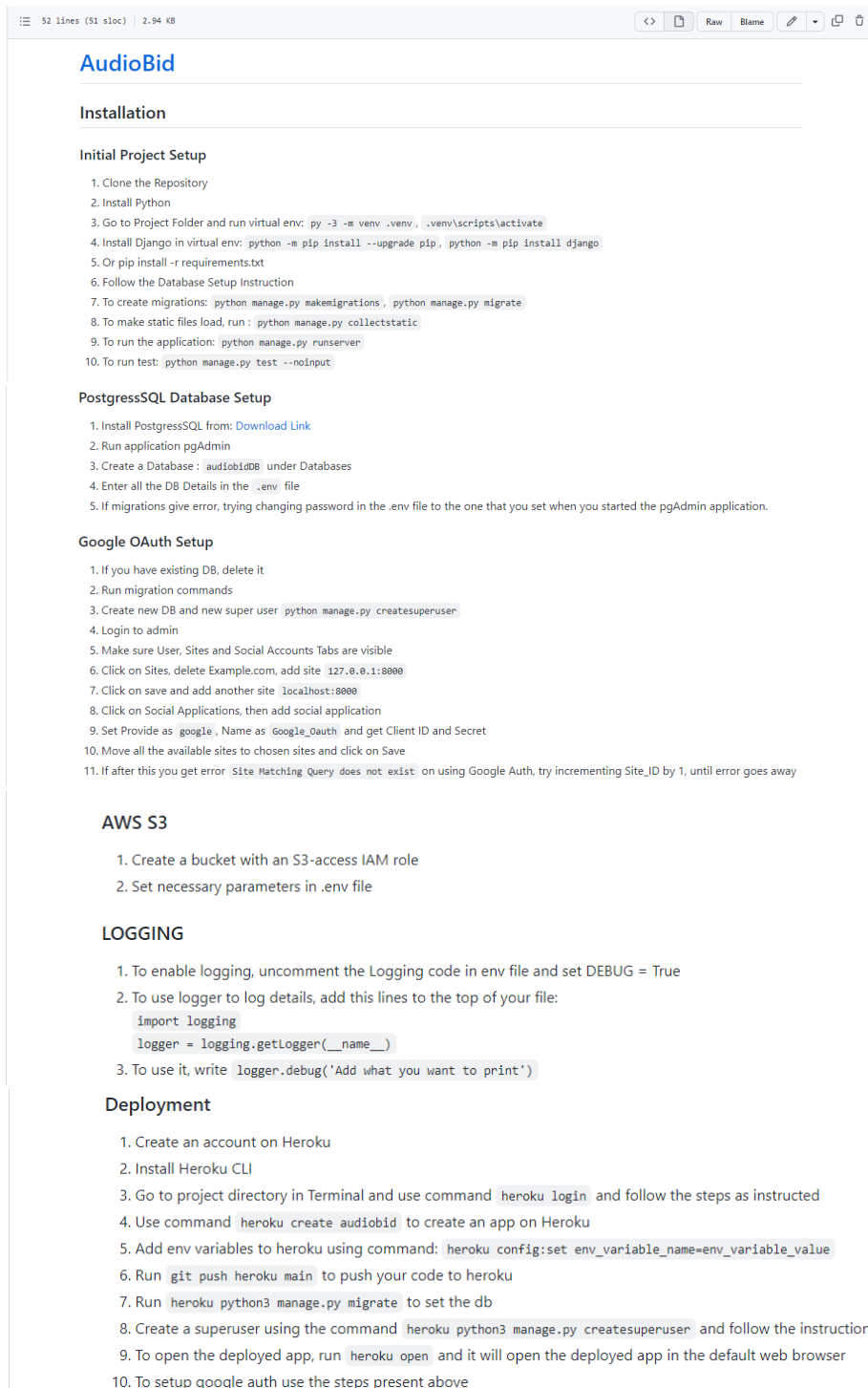
User Stories summary [15 points]

Story Title	Brief description	Story points	Implementation Status	Future changes to the story
Iteration 0 report	A report for the previous iteration.	2	Implemented	The entire project changed.

	Including it here since our project was changed.			
Iteration 1 report	A report for the iteration	2	Implemented	None
Project environment setup on every team member's local machine	A Readme document and initial setup with all steps to get the app running on a user's local machine.	2	Implemented	Extra python packages as and when required. Google Auth setup AWS setup
Create 2 types of users - creator and worker	An initial story for setting up the environment for creator and worker roles.	2	Implemented	Changes in these models as and when new features were added.
Signup, Login and Logout	Signup and login as a creator and worker. Signup requires filling up of	3	Implemented	None
Google signup	Signup using a google account directly.	2	Implemented	A popup notification asking the user to fill extra details required by the portal to maintain database integrity.
Homepage UI	A basic initial homepage	2	Implemented	Lots of changes made based on other features added.

User story screenshots:

(1) Project environment setup on local machines



The screenshot shows a README file for a project named "AudioBid". The file is viewed in a code editor with a light blue header bar showing "52 lines (51 sloc) | 2.94 KB". The README is organized into sections: "Installation", "Initial Project Setup", "PostgreSQL Database Setup", "Google OAuth Setup", "AWS S3", "LOGGING", and "Deployment". Each section contains a list of numbered steps with code snippets highlighted in a light blue background.

AudioBid

Installation

Initial Project Setup

1. Clone the Repository
2. Install Python
3. Go to Project Folder and run virtual env: `py -3 -m venv .venv, .venv\scripts\activate`
4. Install Django in virtual env: `python -m pip install --upgrade pip, python -m pip install django`
5. Or pip install -r requirements.txt
6. Follow the Database Setup Instruction
7. To create migrations: `python manage.py makemigrations, python manage.py migrate`
8. To make static files load, run: `python manage.py collectstatic`
9. To run the application: `python manage.py runserver`
10. To run test: `python manage.py test --noinput`

PostgreSQL Database Setup

1. Install PostgreSQL from: [Download Link](#)
2. Run application pgAdmin
3. Create a Database : `audiobiddb` under Databases
4. Enter all the DB Details in the `.env` file
5. If migrations give error, trying changing password in the `.env` file to the one that you set when you started the pgAdmin application.

Google OAuth Setup

1. If you have existing DB, delete it
2. Run migration commands
3. Create new DB and new super user `python manage.py createsuperuser`
4. Login to admin
5. Make sure User, Sites and Social Accounts Tabs are visible
6. Click on Sites, delete Example.com, add site `127.0.0.1:8000`
7. Click on save and add another site `localhost:8000`
8. Click on Social Applications, then add social application
9. Set Provide as `google`, Name as `google_oauth` and get Client ID and Secret
10. Move all the available sites to chosen sites and click on Save
11. If after this you get error `Site Matching Query does not exist` on using Google Auth, try incrementing Site_ID by 1, until error goes away

AWS S3

1. Create a bucket with an S3-access IAM role
2. Set necessary parameters in `.env` file

LOGGING

1. To enable logging, uncomment the Logging code in env file and set `DEBUG = True`
2. To use logger to log details, add this lines to the top of your file:

```
import logging
logger = logging.getLogger(__name__)
```
3. To use it, write `logger.debug('Add what you want to print')`

Deployment

1. Create an account on Heroku
2. Install Heroku CLI
3. Go to project directory in Terminal and use command `heroku login` and follow the steps as instructed
4. Use command `heroku create audiobid` to create an app on Heroku
5. Add env variables to heroku using command: `heroku config:set env_variable_name=env_variable_value`
6. Run `git push heroku main` to push your code to heroku
7. Run `heroku python3 manage.py migrate` to set the db
8. Create a superuser using the command `heroku python3 manage.py createsuperuser` and follow the instructions on screen
9. To open the deployed app, run `heroku open` and it will open the deployed app in the default web browser
10. To setup google auth use the steps present above

(2) Login, Signup and Logout, Google Authentication

The signup functionality consists of a form that takes in all the basic information about the user. The user has to also choose whether he wants to sign up as a “creator” or “worker”. The user is redirected to his/her homepage directly after signup.

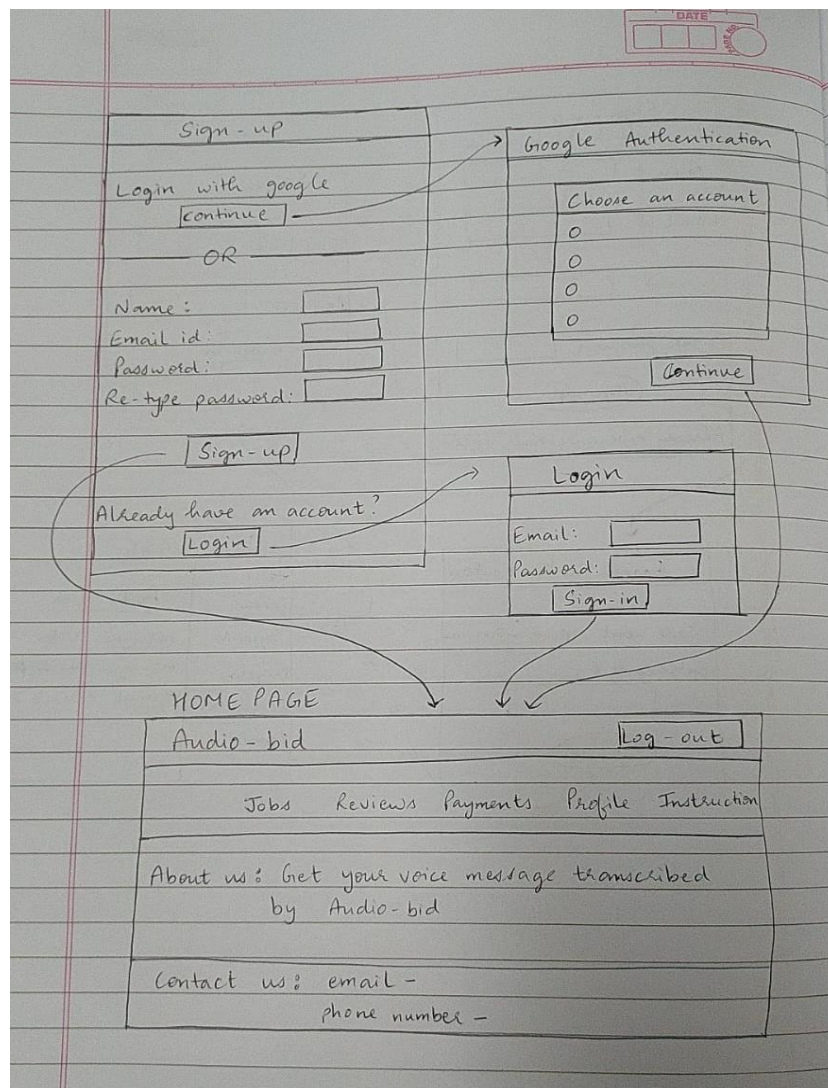
If using Google signup, then the user is required to login into his/her gmail account and later fill a popup notification form asking for the missing pieces of information compared to the traditional sign up.

Login - Takes in an email ID and a password from the user. There can be 2 error cases in this scenario:


1. Invalid username - Notification to tell the user to sign up first.
2. Invalid password - Incorrect password notification.

Backend Details:

We have used a model "Profile" with a "user ID" as the primary key. The table consists of the role (creator/worker), time zone of the user and whether the account has been created with normal or google authentication.




Sign In

 Sign in with Google


OR

[Forgot your password?](#)



[Don't have an account yet? Sign Up](#)


Sign Up

 Sign up with Google

OR

Time zone: UTC

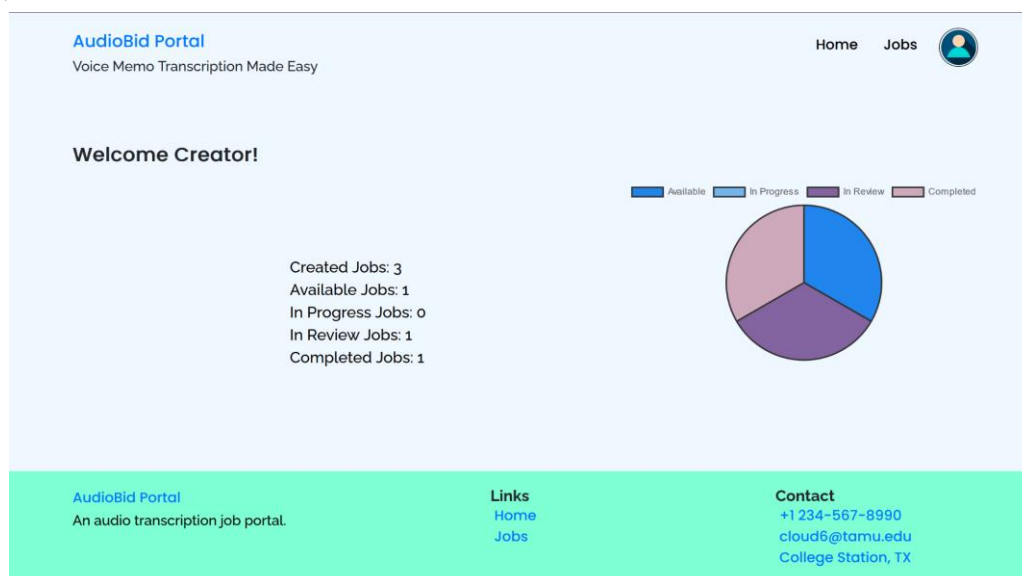
User Type: ☒ Worker ☐ Creator



[Already have an account? Login](#)

(3) Homepage UI

The homepage displays a pie-chart of the jobs status of the user (This functionality was added in a later iteration). In addition, there is a “Jobs” hyperlink in the header that takes the user to the jobs page. The footer consists of the basic information of the application along with the hyperlinks. It also contains a profile/user picture that allows the user to view his/her profile and logout.



Iteration 2 [PO : Ajinkya Bharat Malhotra SM : Vasu Naman Verma]

At the start of this iteration, we had our login feature working along with a basic homepage. This iteration mainly focused on creating a model/schema for storing job details in the database. The client's main requirements for this iteration was to be able to upload jobs as a creator and view the created jobs.

Client Meeting Summary [10/21/2022]

Demo of the previous iteration	Current Iteration requirements
<ul style="list-style-type: none">• A basic version of the application deployed on Heroku• Client tried to sign up as a creator and login using the same. Tried the Google auth functionality too.	<ul style="list-style-type: none">• Minor UI changes to the previous iteration stories.• Creator to be able to create a job by uploading an audio file from his/her local machine.• The job must be visible in the jobs page.• A profile page for the user to be able to see his details.• Forgot password functionality

User Stories summary [9 points]

Story Title	Brief description	Story points	Implementation Status	Future changes to the story
Iteration 2 report	A report for the iteration	2	Implemented	None
Minor UI changes and cleanup along with starting of "Forgot password" functionality	A cleanup story for finalising the existing stories. Details described below. A base for the "Forgot password" feature.	3	Implemented	"Forgot password" completed later on.
Job Model	Create a job model on Django models with appropriate key settings	1	Implemented	Incorporated certain fields like "priority", "status" etc. later on.
Signup logic change	According to the client's request, on signUp, the user	1	Implemented	None

	should be logged in to the website and it should display successfully logged in.			
Popup for extra information in case “Profile” DB is incomplete	An extra popup asking the user to fill additional details that are used to maintain the integrity of the “Profile” model.	2	Implemented	None

User story screenshots:

(1) Cleanup - Profile model, Login, Signup and “Forgot password” initial changes

1. SignUp - A new detailed form was created with the user specifying himself/herself as a creator or worker.
2. Login - Error and success message popups
3. Google Auth - Logo change to make it user friendly.
4. Profile model - Updation of the “Profile” table.
5. Forgot password - An initial page with a “Reset password” button and a form asking for the user’s email address was created. However, no email is sent at this stage.

(2) Job model

The job model consists of the following fields:

1. Name - A title specified by the creator of the job.
2. User ID - A foreign key to synchronise the job model with the profile model. It corresponds to the creator who has created the job.
3. Price - The initial price of the job set by the creator.
4. Limit_price - The maximum value up to which the price can increase dynamically.
5. Created_date, End_date - The dates on which the job was created and a deadline.
6. Cron_date - Used to keep track of the dynamic price increase.
7. Claim_date - The date on which the job was claimed by a worker
8. Description - The job description specified by the creator.
9. Url2audio, Url2transcript - A url of the location where the audio file and transcripts are stored. Updated in the backend.
10. Worker_id - The ID of the worker who has claimed the job.
11. Content - This field contains the transcript written by the worker.
12. Status - This field is used to indicate the status of a job.
 - Available - A job that has been created and not been picked up by any worker
 - In progress - A job actively being worked on by a worker
 - Completed - A job that has been reviewed by the creator and marked as completed when he/she is fully satisfied with it.
 - In review - A job being reviewed by the creator after submission from the worker’s side.

```

class Job(models.Model):
    name= models.CharField(max_length=50, null=True)
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    price = models.DecimalField(max_digits=19, decimal_places=10)
    limit_price = models.DecimalField(max_digits=19, decimal_places=10, default=0)
    created_date = models.DateTimeField(auto_now_add=True)
    end_date = models.DateTimeField()
    cron_date = models.DateTimeField(null=True)
    claim_date = models.DateTimeField(null=True)
    description = models.TextField()
    url2audio = models.TextField()
    url2Transcript = models.TextField(null=True)
    worker_id = models.CharField(max_length=100, default='0')
    content = models.TextField()
    status_choices = [ # 0 = not started, 1 = in progress, 2 = completed, 3 = cancelled
        (0, 'AVAILABLE'),
        (1, 'INPROGRESS'),
        (2, 'COMPLETED'),
        (3, 'INREVIEW')
    ]
    status = models.IntegerField(choices=status_choices, default=0)

```

(3) Popup form

When signing up using Google O-Auth, the user's email address and password gets updated to the database. However, we have created a separate popup after login to collect the remaining details required by the "Profile" model.

We have extended this feature in such a way that the popup would be displayed in all those cases when there is an inconsistency with the "Profile" model.

Please fill out the missing info below.

Finish Profile

Vasu Verma

vasuverma

vasuverma@tamu.edu

Time zone: US/Central

User Type:

☐ worker ☒ creator

Save Changes

Iteration 3 [PO : Neel Akash Murugesan Sathya Bama SM : Swatthi Vijay Sanker]

Iteration 3 focused on the creation of jobs, viewing them on both the creator and worker sides along with certain final clean-up changes and unit tests for sign up and login functionalities. Create job was modified to incorporate 3 methods of providing audio - On the spot recording option, uploading an audio file or providing a google drive URL of audio.

Client Meeting Summary [10/28/2022, 11/04/2022]

Demo of the previous iteration	Current Iteration requirements
<ul style="list-style-type: none">• Signed in using the creator account created by him in Iteration 2.• Created a job successfully by uploading an audio file from his local machine.• Was able to view all his profile details entered during signup in the profile page.	<ul style="list-style-type: none">• Further minor changes in UI and functionality in Sign up and login pages.• To be able to create audio by recording on the spot and providing a google drive URL.• View the created jobs along with their status in his jobs page.• Login as a worker and be able to see the set of available jobs.

User Stories summary [17 points]

Story Title	Brief description	Story points	Implementation Status	Future changes to the story
Iteration 3 report	A report for the iteration	2	Implemented	None
Forgot password	Forgot password functionality to enable the user to reset his/her password.	3	Implemented	None
Homepage cleanup (Minor story)	The homepage UI was modified in accordance with the client requirements.	1	Implemented	None
Upload audio - 3 options	Creator should be able to create audio in 3 ways:	3	Implemented	Incorporated pause and resume feature

	Browse a locally stored file. (OR) Record audio on the spot (OR) Provide a google drive URL to the audio.			to on-the spot recording.
Create UI for Profile tab	View my profile information as a user.	2	Implemented	Option to edit certain fields in the profile page.
Create Job Posting UI	A frontend popup was implemented that would be displayed when the creator tries to create a job.	2	Implemented	Changes to the form including “Limit price” etc. were made later on.
Create UI for creator’s job tab	The creator should be able to view the jobs that he/she has created on the jobs page. The worker should be able to view the list of available jobs and the jobs claimed by him/her.	2	Implemented	Added the sort and filter functionalities.
Unit tests	Unit test cases were written to verify the functionalities implemented so far.	2	Implemented	None

User story screenshots:

(1) Forgot password


A user can use this functionality to get an email link to reset his password. We also take care of security here by authenticating the entered email address with the database. If the email address is not registered, we do not allow the user to reset the password.


Reset Password

Forgotten your password? Enter your email address below, and we'll email instructions for setting a new one.

Email: Send email

Forgot password email sent using Amazon SES which contains a link to the reset password page, which only be used once.

Password Reset Requested 

 **audiobidservice@gmail.com** via amazonses.com
to me ▾

3:54 PM (0 minutes ago) ☆ ↶ ⋮

Hello,

We received a request to reset the password for your account for this email address. To initiate the password reset process for your account, click the link below.

<https://audiobid.herokuapp.com/reset/Nw/bg6q6c-2f5518374fb16d736195bff5f4ae4506/>

This link can only be used once. If you need to reset your password again, please visit <https://audiobid.herokuapp.com> and request another reset.

If you did not make this request, you can simply ignore this email.

Sincerely,
The Audio Bid Team

↶ Reply

↷ Forward

Links provided in the email takes us to the Reset password Page to create a new password for the user.

Password Reset

Please enter your new password.


New password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

New password confirmation:

Reset password

(2) Create UI for Profile tab



Bhavya
bhavyarshah1208@gmail.com

First Name:

Last Name:

Username:

User Role:

Email ID:

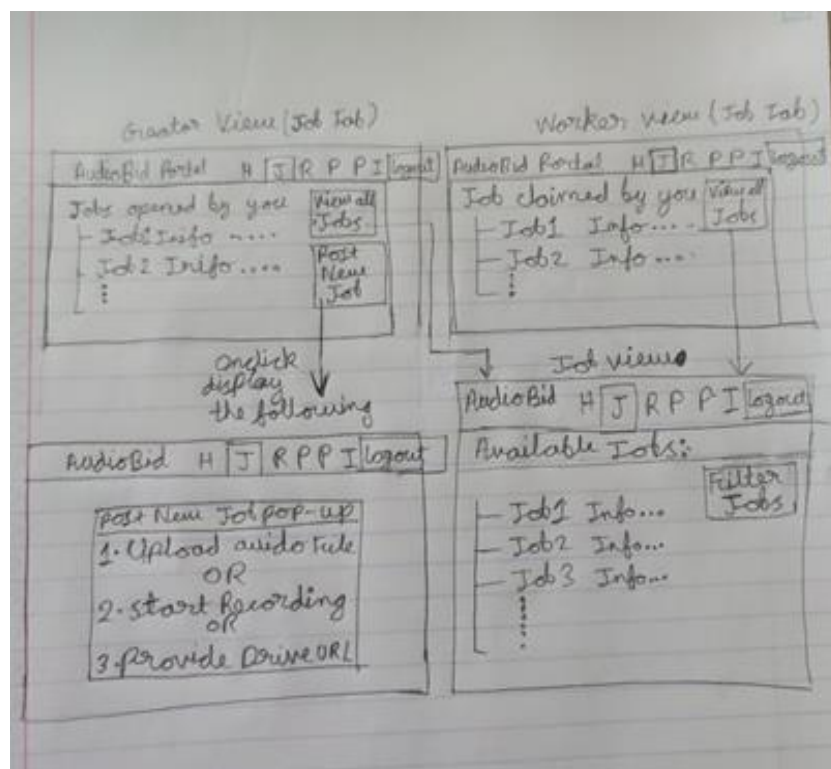
Time zone:

[Save Changes](#)

The user is only allowed to change the time zone. This was implemented as per client request.

(3) Create job posting UI

The creator can create a job with 3 options (included in a later iteration). A popup comes up asking the user to enter the details of the job and browse the desired audio file. The backend code handles the saving of the audio file to AWS S3 and updating the file's URL to the database.



Provide Job Details

Title:

Description:

Job Price:

Limit Price:

End date: 12/09/2022

Choose Audio Option

Save

(4) Unit tests:

Tests.py include 9 unit tests, which checks the models as well as some of the views.

To check models, we are creating a user, updating the profile, creating a job, review and comment. We are checking values in db with what we used to create, to make sure nothing is changing in DB during the creation process.

To check views, we login using a user and checking if the user is authenticated or not. We then get the views and check if the https status code is 200 and the page rendered is what we expect to render and if the html appeared contains the text we expect.

```
C:\TAMU\First SEM\Software Engineering\Audio-Bid>python manage.py test --noinput
Found 9 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
.....
-----
Ran 9 tests in 0.978s

OK
Destroying test database for alias 'default'...

C:\TAMU\First SEM\Software Engineering\Audio-Bid>
```

(5) Audio options

A purely backend story wherein we have provided the creator with 3 options to create an audio file. Either he/she can browse a file from their local machine or record audio on the spot or provide a google drive URL. The URL then gets updated to the Job model to maintain consistency.

(6) View created jobs

Creator - The display page consists of only the jobs created by him/her. In the backend, a filter with “request.user.id” is applied to fetch the jobs pertaining to the user.

Worker - The display consists of the jobs claimed by the worker along with the list of available jobs (by all creators for the worker to claim). We make sure that jobs claimed by other workers are not displayed here. In the backend, the jobs are filtered using the “worker id” parameter in the Job model.

Id ↑ ↓	Title ↑ ↓	Price ↑ ↓	Created on ↑ ↓	Status ↑ ↓
1	New Test	122.00	Dec 2, 2022	COMPLETED
2	test	10.00	Dec 2, 2022	COMPLETED
4	Need help to convert a conference speech to text	100.00	Dec 7, 2022	AVAILABLE

Iteration 4 [PO : Abhiraj Singh Rathore SM : Bhavya Rushin Shah]

At the start of Iteration 4, most of the basic functionalities were completed. This iteration focused on enabling the creator and worker to download audio files, transcripts and most importantly review the jobs completed. As an additional feature, an online text editor was also included to allow a worker to type out his/her transcript online. The sort and filter feature was also completed along with dynamic pricing.

Client Meeting Summary [11/18/2022]

Demo of the previous iteration	Current Iteration requirements
<ul style="list-style-type: none"> Signed up using the Google auth functionality and was able to fill and 	<ul style="list-style-type: none"> The creator should be able to review the work completed by a worker and

<p>save the additional popup form.</p> <ul style="list-style-type: none"> • Tried out all the 3 options of storing audio successfully. • Viewed all the jobs as both creator and worker along with being able to delete and edit the job as a creator. • Was able to claim a job as a worker and notice its status change to “IN PROGRESS” from “AVAILABLE”. 	<p>provide review comments.</p> <ul style="list-style-type: none"> • In order to review a job, the creator should be able to download the transcript file provided by the worker. The audio file should also be accessible. • Filter jobs according to various parameters like “created date”, “end date”, “priority” and “price”. • The price should be incremented in regular intervals until it has been claimed by a worker.
---	---

User Stories summary [43 points]

Story Title	Brief description	Story points	Implementation Status	Future changes to the story
Iteration 4 report	A report for the iteration	2	Implemented	None
Script for in-class presentation	Prepared a script for the in-class demo.	2	Implemented	None
Job posting form verification	Added scripts to verify the details entered by the user in the Create job form.	2	Implemented	None
Homepage UI pie-chart	A pie-chart displayed on a user’s homepage to indicate the progress so far.	2	Implemented	None
Upload and download transcript, audio for worker and creator	A worker should be able to upload a transcript and audio when he/she has completed the job. The creator should be able to download it.	3	Implemented	None
Create UI for job editing, Job editing	A creator should be able to edit the job created by him.	3	Implemented	None

functionality	He/she can change the “deadline”, “price” of the job etc.			
UI to display jobs	A final story to display the appropriate jobs for both creators and workers. Shifted the display of details to a separate page handled by another story	2	Implemented	None
Create UI for viewing posted job	A UI for displaying the details of a job when clicked upon from the job table. It displays the description, price, dates etc. in a separate page.	2	Implemented	None
Create Job UI changes	New UI for the create job modal. The earlier UI was similar to the one on LinkedIn that had to be changed.	2	Implemented	None
Sort and Filter	Filter the jobs table based on created date, end date, price and status fields. Sort the table in ascending and descending order based on each column of the table.	3	Implemented	None
Online text editor for transcripts	A worker can type out his/her transcripts in an online editor instead of uploading a file.	3	Implemented	None
Accept/Discard job	A creator should be able to discard a job if not satisfied which would put back the job into the available pool.	2	Implemented	Added review and rating features.

	He/She can accept a job and provide a star rating out of 5 to the worker who has completed the job.			
Merge and deployment - Start to finish	Final merge of all the features up to iteration 4 and deployment to Heroku along with behavioural testing	3	Implemented	None
Logging for development	The debugging flag can be set to true in the settings.py file during development, after which all the events, DB queries, and errors will be outputted to the debug.log file. The flag is set to false for the production env. This enabled the team to figure out mistakes easily and made the workflow more efficient.	1	Implemented	None
Profile dropdown	A drop down on the profile icon to display the rating of the user.	2	Implemented	None
Audio recorder: Pause and resume recordings, update job model	Incorporated the pause and resume feature to audio recording mentioned in Iteration 3. Updated the job model with proper fields to avoid inconsistencies	2	Implemented	None
Worker - Claim job	A worker should be able to claim any	3	Implemented	None

Creator - Delete and edit a job	<p>available job to start working on it.</p> <p>A creator should be able to delete and edit a posted job.</p>			
Review a job as a creator and view reviews as a worker	A creator can provide review comments to a submitted job. If not satisfied, he/she can mention it and request the worker to reupload another transcript. The worker should be able to view these comments.	2	Implemented	Upgraded the feature to a review conversation/thread similar to a mini-chat feature.
Rate the job as a creator	Provide a rating out of 5 stars for any job completed by a worker.	2	Implemented	Display the same on a worker's profile page.

User story screenshots:

(1) Upload and download audio transcripts, Online Text editor

A worker can either upload the audio transcript files to our system or type out the transcript using the online text editor. There is also a “Save Transcript for later” option that allows a user to save the transcript and continue working on it later on.

Speciality of the text editor feature - This feature ensures that any worker irrespective of his/her internet connection speeds can make use of the Audio Bid portal to earn money. In case of low internet bandwidth, the worker does not have to download the huge audio files. He/she can listen to it online and type out the transcript in the online text editor that makes it easier.

Job Details

Job Title: New Test

Job Description:
It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

Job Price: \$122.00

Created on: Dec. 2, 2022, 7:30 a.m.

Due Date: Dec. 20, 2022, midnight

Job Status: COMPLETED

Download Audio File
Download Transcript File

Job Details

Job Title: Need help to convert a conference speech to text

Job Description:
 It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

Job Price: \$100.00

Created on: Dec. 7, 2022, 3:09 p.m.

Due Date: Dec. 30, 2022, midnight

Job Status: in progress

File Edit View Insert Format Tools Table Help

Paragraph
 B I

This domain is not registered with Tiny Cloud. Please see the [quick start guide](#) or [create an account](#).

0 words tiny

Save Transcript for Later

Download Audio File


Upload transcript


Cancel Job


(2) Profile dropdown

In the current iteration, we created a dropdown that lets a user edit his/her profile and Log out of the portal. The star rating shown here was completed in a later iteration.

Jobs




Worker

 4.0

Edit Profile

Log Out

(3) Sort and Filter

We have firstly used a paginator to display the jobs in the “jobs” page. There is an “Apply filter” option at the top right corner that opens a sidebar asking for the filter settings. The user can input the desired filter values and even clear all the filter fields. In addition to filter, there is a sort functionality that sorts the table in ascending or descending order based on each column of the table displayed.

The image shows a web application for a job portal. A modal titled "Filter tab" is open, allowing users to filter jobs by "Created Date", "End Date", "Price" (with MIN and MAX inputs), and "Status". Below the filters are "Clear Filter" and "Apply Filter" buttons. In the background, a table of jobs is visible with columns for "Id", "Title", "Price", "Created on", and "Status". A table header bar at the bottom of the page includes sort arrows for each column and an "Apply Filter" button on the right.

(4) Accept, rate and discard a job

Accept - A creator can accept a job submitted by a worker. This changes the job’s status to completed that indicates that the job is complete.

Rate - A creator can rate a worker based on his/her satisfaction. This provides every worker with a star rating. As a future scope of this project, a bidding system can be implemented that decides a job’s price specific to every worker based on the star rating.

Discard - A creator has the option to discard a job if not satisfied which puts the job back into the available pool. This indicates that the job’s status is now “available” and any worker can pick it up.

The image shows a "Job Details" page. At the top, the job title is "Need help to convert a conference speech to text", with "Accept Job" and "Discard Job" buttons. The "Job Description" is a paragraph of Lorem Ipsum text. Below this, the "Job Price" is \$100.00, "Created on" is Dec. 7, 2022, 3:09 p.m., and "Due Date" is Dec. 30, 2022, midnight. The "Job Status" is "INREVIEW". At the bottom, there are three buttons: "Download Audio File", "Download Transcript File", and "Review Job".

(5) Reviews

A creator can use the “Comments” option to provide review comments for an “in review” job. These comments would be displayed to the worker working on this job. In the backend, there is a separate model named “ReviewRating” that keeps a record of the creator id and worker id (to associate it with the Job model), review and rating fields along with the date and time of creation. This helps display the review comment to the corresponding worker.

Job Review

Job Creator said

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words

Comment(s) 0

Worker →

Job Details

Job title :

Job Description :

Job price :

Created on :

Due Date :

Download File

Upload Transcript

Your Review

Job Review

Creator's Comments :

Your Comment :

(6) Claim, delete and edit jobs

Creator - Every job in the table contains a hyperlink to a new page wherein all the details are displayed. A creator can edit certain fields such as the title and the price (to enable dynamic pricing) of the job. He/she can also delete the job as long as it is in available status.

Worker - A worker can use the “Claim Job” feature to assign a job to himself. In the backend, we change the status of the job to “in progress” and assign the worker’s ID to the job model. By changing the status, we ensure that no other worker can pick the same job.

Job Details

Job Title: Need help to convert a conference speech to text

Edit Job

Job Description:

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

Job Price: \$100.00

Created on: Dec. 7, 2022, 3:09 p.m.

Due Date: Dec. 30, 2022, midnight

Job Status: AVAILABLE

Download Audio File

Delete Job

Job Details

Job Title: Need help to convert a conference speech to text

Job Description:

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

Job Price: \$100.00

Created on: Dec. 7, 2022, 3:09 p.m.

Due Date: Dec. 30, 2022, midnight

Job Status: AVAILABLE

Claim Job

(7) Home page UI - Pie chart

When a user login is complete, we display a pie-chart indicating the current progress. For a creator, it displays the status of the jobs created by him - the number of jobs still unassigned, number of jobs completed etc. For a worker, it displays the status of his/her jobs. This includes the number of jobs claimed by him and in progress along with the number of jobs he/she has completed so far.

(8) Logging functionality

```
194 # Logging - Uncomment when using development env and set DEBUG to True
195 # LOGGING = {
196 #     'version': 1,
197 #     'disable_existing_loggers': False,
198 #     'handlers': {
199 #         'file': {
200 #             'level': 'DEBUG',
201 #             'class': 'logging.FileHandler',
202 #             'filename': BASE_DIR / 'debug.log',
203 #         },
204 #     },
205 #     'loggers': {
206 #         '': {
207 #             'handlers': ['file'],
208 #             'level': 'DEBUG',
209 #             'propagate': True,
210 #         },
211 #         'django': {
212 #             'handlers': ['file'],
213 #             # Change it to INFO if you donot want to output everything
214 #             'level': 'INFO',
215 #             'propagate': True,
216 #         },
217 #     },
218 # }
```

Iteration 5 [PO : Swatthi Vijay Sanker SM : Neel Akash Murugesan Sathya Bama]

In the last iteration, the main focus was on performing tests and checking if all the functionality worked as expected and ensuring that the project can be continued by another team at a later point of time. This included adding comments to the codebase and creating AWS credentials for the client to be able to host the application on Heroku on their own. Minor changes to 2 of the stories have also been done in this iteration.

Client Meeting Summary [12/02/2022]

Demo of the previous iteration	Current Iteration requirements
<ul style="list-style-type: none">Final demo of all the functionalities of the website. The client was satisfied with all the creator and worker features.	<ul style="list-style-type: none">Since this was the last iteration, the client had no specific requirements. As a future scope, these 2 features are to be added:<ul style="list-style-type: none">a. Payments handling - Using Stripe APIb. Bidding system - As of now, we have implemented dynamic pricing which keeps incrementing the price every 24 hours. In the future, a dynamic bidding system is expected where workers can bid for jobs and workers with higher ratings get better pricing.

User Stories summary [18 points]

Story Title	Brief description	Story points	Implementation Status	Future changes to the story
Iteration 5 report	A report for the iteration	2	Implemented	None
Add comments to the codebase	Describe the functionality of each part of the code so that a team working on this project in the future can do so seamlessly.	2	Implemented	None
Create AWS and Heroku credentials	Create these credentials for the client to be able to host the application of his/her own.	2	Implemented	None
Review threads/conversations	Modified the reviews feature to become a dynamic mini-conversation	2	Implemented	None
View average rating in profile page	A worker can view his/her average rating in the profile dropdown	2	Implemented	None
Dynamic job pricing	The price set by a creator for a job should be automatically incremented by \$1 every 24 hours either until it reaches the limit price or a worker has claimed it.	2	Implemented	None
Migrations reduction	Reduced 27 migrations file to a single one so that it is easier for the next	1	Implemented	None

	team			
Final report	A final report as per the guidelines.	3	Implemented	None
Final Video	2 final videos for the website demo and the presentation as per the guidelines.	2	Implemented	None

User story screenshots:

(1) Add comments to codebase

(2) Create AWS and Heroku credentials.

We are using AWS S3 to store the audio files and transcripts uploaded by the users and Heroku to deploy the website. Currently, we are using our personal accounts to do so. We have migrated the same to the client's account for any team to continue this project in the future.

(3) Review threads/conversations

The review feature in Iteration 4 allowed only the creator to add review comments and the worker to view the same. However, in this iteration, we have modified it to a mini-conversation allowing both the creator and worker to talk to each other in a chat format. In the backend, we have created another model "Comment" that maps to a particular job. This is updated each time a new comment is added to a job by either a worker or a creator.

(4) View average rating in Profile page

The profile dropdown feature in Iteration 4 displayed an option to "Edit profile" and to Log out of the application. In this story, we display an average star rating of a worker logged in.

In the backend, we have added 2 fields namely "Rating" and "Number of ratings" to the Profile model that keeps track of the average rating of a worker.

(5) Dynamic job pricing

When a creator creates a job, they have to enter a "limit price" for the job created. The price of a job is dynamically incremented up to the limit price. In the backend, we use the "Cron job" package to query the database every 6 hours to check if 24 hours has elapsed since the time of job creation. If yes, it automatically increments the price of the job by \$1 until the limit is reached or a worker has claimed the job. If the job price has reached the limit price, then it will remain the same as long as the job remains in the available job pool.

BDD/TDD Process:

BDD (Behaviour Driven Development) is an approach where we test the actual behaviour of the system from the end users perspective. We used the BDD approach to write tests in simple language in accordance with the user requirements to test after the entire functionality is done, instead of writing test code in between. The entire process can be detailed as:

1. First, we discussed the behaviour of the software and the features which are to be built.
2. We developed code by testing sample data in a continuous manner through communication with the client.
3. For each feature, we first got requirements from the client and then tested the code based on examples from the different users' perspective.

In the last iteration, we wrote unit test cases as well.

Advantages:

1. Being a nontechnical process by nature, it was much easier to understand and convey to the client.
2. We clearly communicated the requirements with the client so there was less wastage or rework due to misunderstood functionalities.
3. We could develop code that can be actually used in business.

Disadvantages:

1. The requirements developed and changed during the progress of the development, and so in the beginning there were some misinterpreted features that had to be reworked on.
2. Communication between the team members was crucial and this caused problems at times.

Configuration Management:

We used GitHub as a configuration management tool for the project. The changes were not merged directly onto the main branch. Instead, every team member created a new development branch to push their code changes. Then he/she raised a pull request before merging the change into the main branch. In total, we had 40 pull requests over the course of 5 iterations. We did not do any spikes.

We did a total of 6-7 releases to Heroku, once after every iteration development was completed.

Issues with Heroku:

- 1) We faced certain problems with migration and creation of the super user and setting the database
- 2) There were issues with static files as they were not working and had to be rectified.
- 3) Finally, there were some issues with setting google authentication with heroku.

Issues with AWS:

- 1) Configuring permissions in S3 Bucket to allow access to a specific AWS IAM user.
- 2) Configuring the same IAM user in python Django app and using its credentials to store and access AWS bucket files

Other tools used:

- 1) WhiteNoise: It allows the web app to serve its own static files, making it a self-contained unit that can be deployed anywhere without relying on nginx, Amazon S3 or any other external service.
- 2) Django-filter: It is a generic, reusable application to alleviate writing some of the more mundane bits of view code.
- 3) Chart.js: It renders chart elements on an HTML5 canvas unlike several other, mostly D3.js-based, charting libraries that render as SVG.
- 4) Django-cron: It lets you run Django/Python code on a recurring basis providing basic plumbing to track and execute tasks.
- 5) Django-SES: It is a drop-in mail backend for Django. Instead of sending emails through a traditional SMTP mail server, Django-SES routes email through Amazon Web Services' excellent Simple Email Service
- 6) Django-allauth: It is an integrated set of Django applications addressing authentication, registration, account management as well as 3rd party (social) account authentication