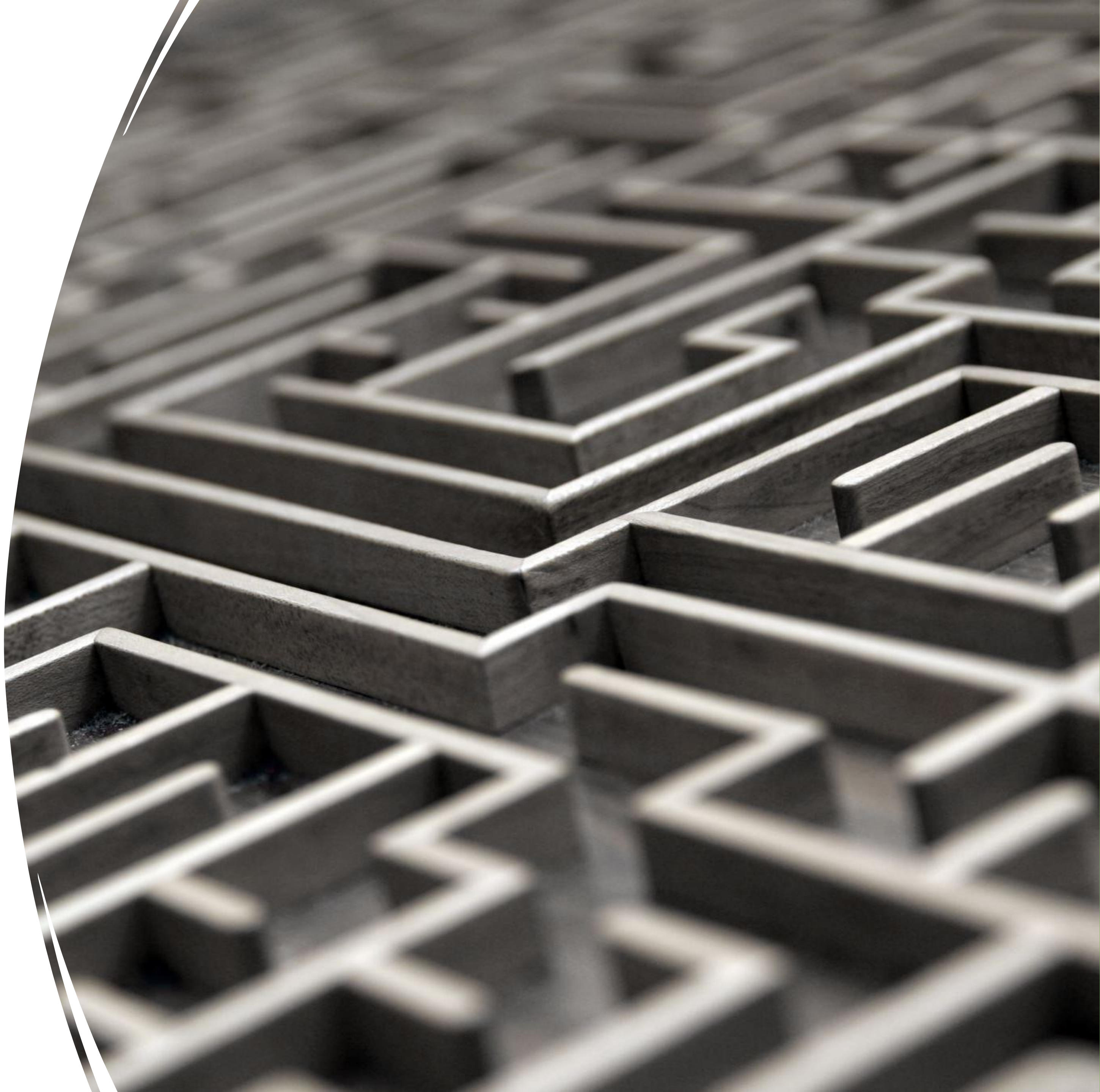
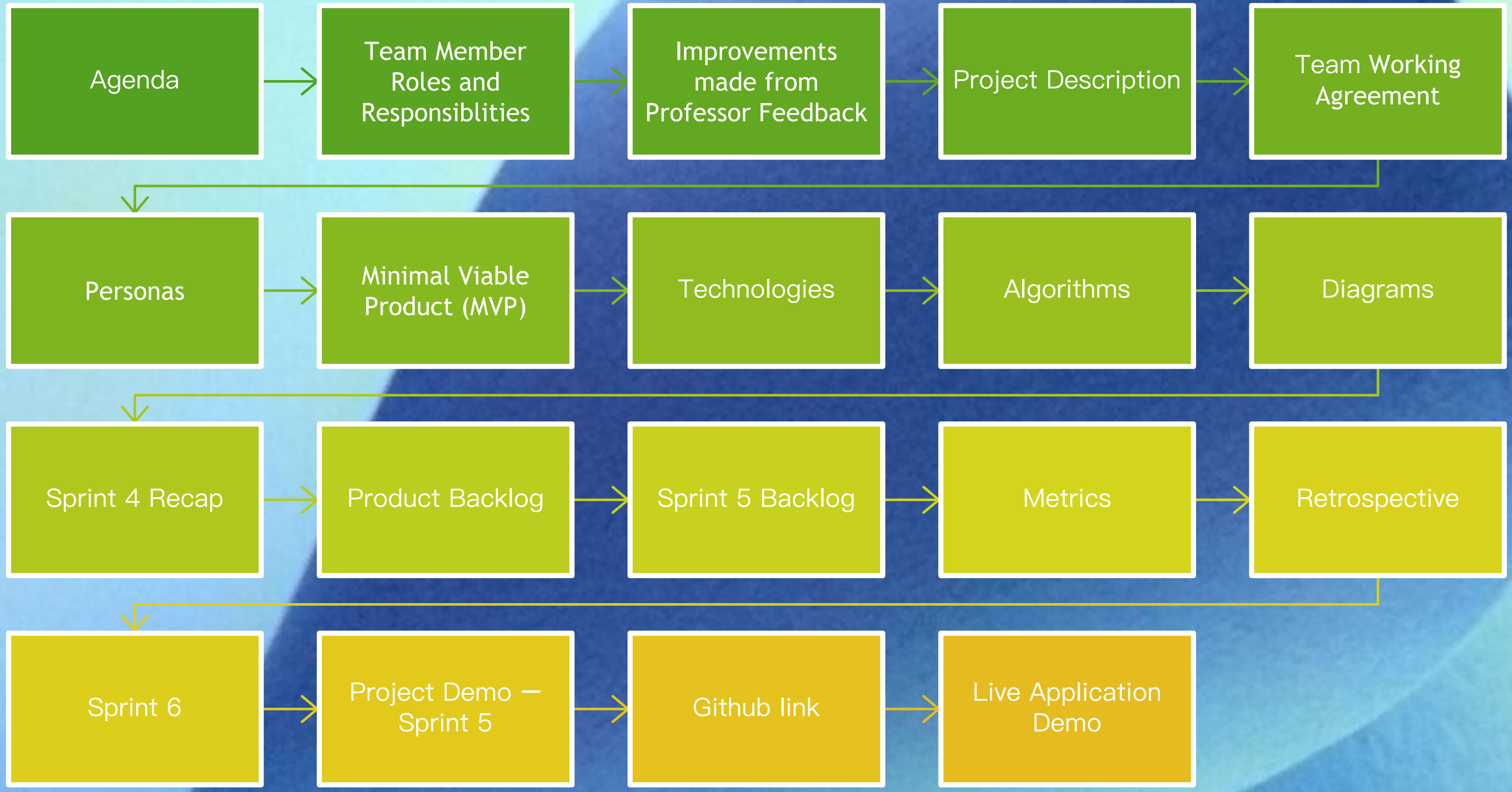
Simply Online

- Simple way to connect



changed

Agenda



Agenda

Team Member Roles and Responsiblities

Improvements made from Professor Feedback

Project Description

Team Working Agreement

Personas

Minimal Viable Product (MVP)

Technologies

Algorithms

Diagrams

Sprint 4 Recap

Product Backlog

Sprint 5 Backlog

Metrics

Retrospective

Sprint 6

Project Demo – Sprint 5

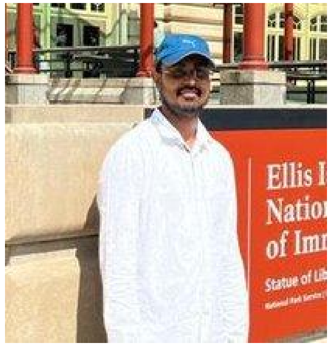
Github link

Live Application Demo

TEAM MEMBERS

Ajay Kumar

Full Stack Developer



Amarendra Reddy

Developer

Pruthvi Raj Reddy

Database Administrator

Mounik

Developer

Sreeja Reddy

Quality Analyst

Ravi Teja Reddy

Developer

Improvement from Professor's feedback

- We incorporated the professor's feedback and make improvements to the presentation.
- Action items in retrospective.
- Average velocity chart in Metrics
- Added the single slide of MVP listing all features and product backlog
- API slide is added in the presentation

changed

Project Description

Simply Online is a web application that aims to simplify the process of online classes. This application allows lecturers to create virtual rooms and share the room id with their students for seamless connectivity. Using webRTC technology, simply online allows group video-sharing features and screen-sharing capabilities. This application allows lecturers to easily take attendance with just a click of a button. This platform includes features like automatic attendance marking and auto transcript generation to eliminate the need for manual tracking and notes taking. Overall, simply online offers a comprehensive and user-friendly solution for educators and students to enhance their online learning experience.

Team Working Agreement

Participation

All the team members are expected to involve in project discussions and attend the meetings promptly. Absence during multiple meetings will affect the team's performance and efficiency.

The team member can discuss beforehand with the team leader if he/she is going to miss the meeting or make it up for it before the next meeting is scheduled.

Communication

The team will communicate with each other using WhatsApp group and meetings will be scheduled on Zoom.

Jira software will be used to track the assigned tasks. For any dependency on another task, mention it in the task comments.

Task management, bugs, sprint planning, and meeting minutes will be tracked in Jira.

To share the final deliverables, Google docs will be used where all the team members can edit the document.

Work Division

The entire project work should be divided into equal parts, and equal responsibilities should be given to all the team members.

Each team member should complete their division of work before the deadline. If they are unable to complete the work on time, that hinders the performance of the entire team. If in case a team member is facing trouble and issues at some point, they can share it with others so that they can help each other and complete the work before the deadline.

Meetings

All the team members will meet on zoom virtually every Tuesday and Friday. All the team members must be present, as attendance is mandatory unless there is an exceptional case.

The team leader would be responsible for sending meeting details and conducting the meeting.

A meeting track or meeting minutes report would be listed after every meeting to keep track of the project and its progress.

Every team member is expected to come up with ideas, participate in the discussion, and give an update on their progress for their part of the work.

TEAM MEMBERS

1. AJAY KUMAR
2. AMARENDAR
3. RAVI TEJA
4. SREEJA
5. PRUTHVI RAJ
6. MOUNIK

Persona

Name Professor James

Age 45

Occupation University Professor

Profile

James is a tenured professor in the Computer Science department at a large university. He teaches both undergraduate and graduate-level courses and conducts research in his field. Due to the COVID-19 pandemic, his classes have been moved online, and he uses various platforms to deliver lectures, holds office hours, and communicate with his students. He lives with his spouse and two children, who are also attending school virtually.

Goals and Motivations

- Deliver high-quality lectures and course material to his students
  Engage his students and create a dynamic and interactive virtual classroom environment.
- Ensure that his students are keeping up with the coursework and meeting their learning objectives.
- Provide effective feedback and support to his students.

Persona

Name Sarah

Age 24

Occupation College student

Profile

Sarah is a full-time student pursuing a degree in psychology. Due to the COVID-19 pandemic, her classes have been moved online, and she uses Zoom to attend lectures, participate in group discussions, and communicate with her professors and classmates.

She lives in a small apartment with roommates and shares a room with one of them.

She has a busy schedule and often has to balance her coursework with a part-time job and other responsibilities.

Goals and Motivations

- Attend all her classes and be an active participant in class discussions
  Stay organized and manage her time effectively to meet assignment deadlines.
- Have a reliable and user-friendly platform for attending virtual classes. Connect with her professors and classmates, and build a community within her course

Persona

Name Ishika

Age 27

Occupation Elementary school teacher

Profile

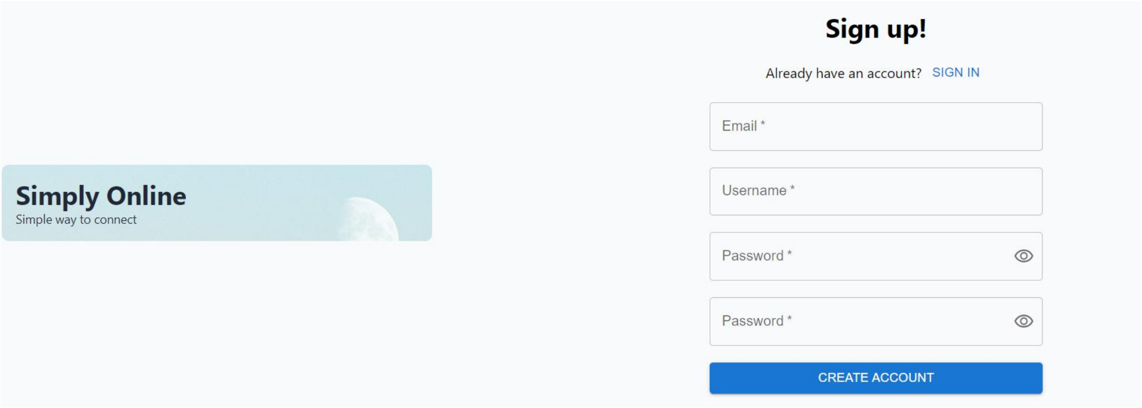
Ishika is a dedicated elementary school teacher who loves working with children. She has been teaching for three years and is always looking for ways to improve her classroom management and student engagement. Ishika is originally from India but moved to the US with her family when she was a child. She is fluent in English and Hindi and enjoys cooking traditional Indian dishes in her free time

Goals and Motivations

- Ishika is a dedicated elementary school teacher who loves working with children. She has been teaching for three years and is always looking for ways to improve her classroom management and student engagement. Ishika is originally from India but moved to the US with her family when she was a child. She is fluent in English and Hindi and enjoys cooking traditional Indian dishes in her free time
- Ishika's main goal is to create a safe and engaging learning environment for her students. She wants to be able to take attendance quickly and efficiently so she can spend more time teaching and less time on administrative tasks. Sarah is also motivated by the opportunity to track student attendance and identify patterns that might indicate a need for additional support.

MINIMAL VIABLE PRODUCT

changed



Simply Online

Simple way to connect

Sign up!

Already have an account? SIGN IN

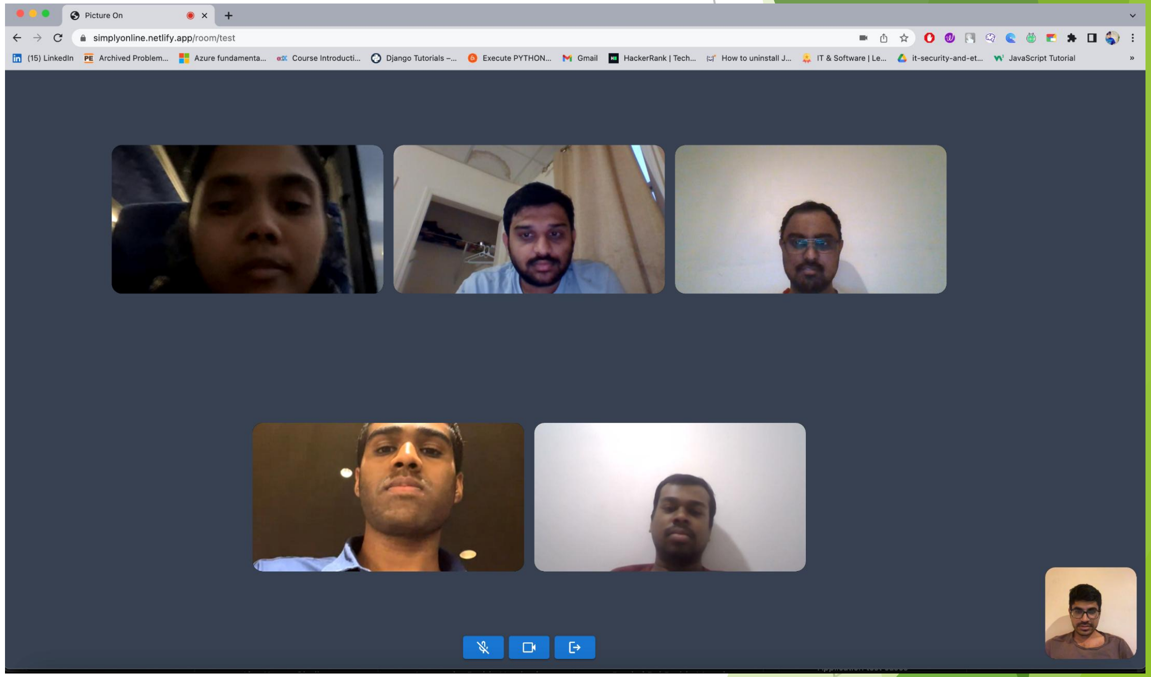
Email *

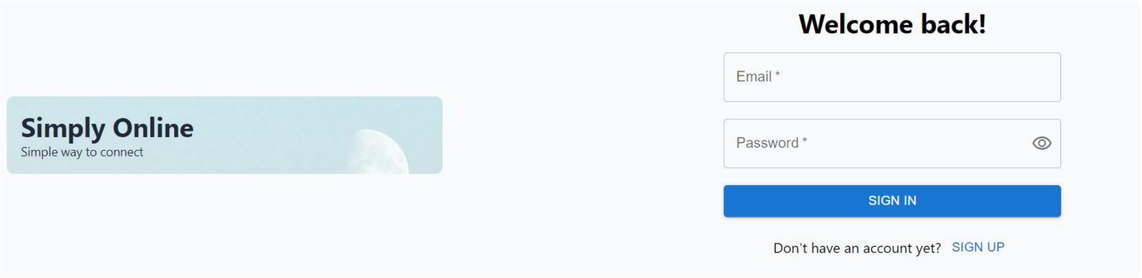
Username *

Password *

Password *

CREATE ACCOUNT





Simply Online

Simple way to connect

Welcome back!

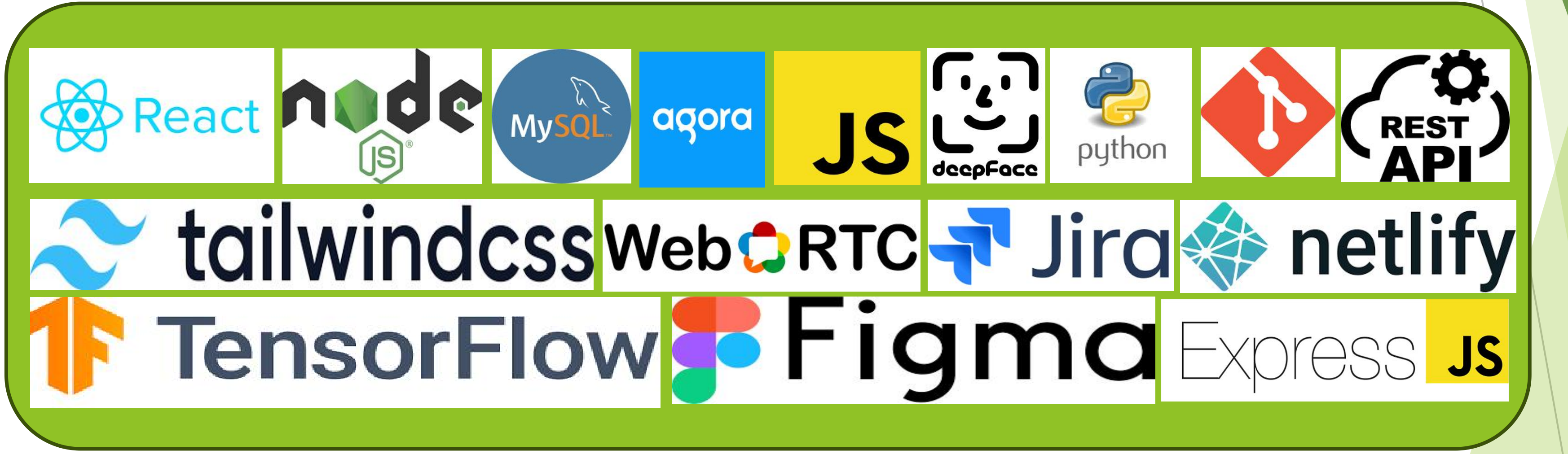
Email *

Password *

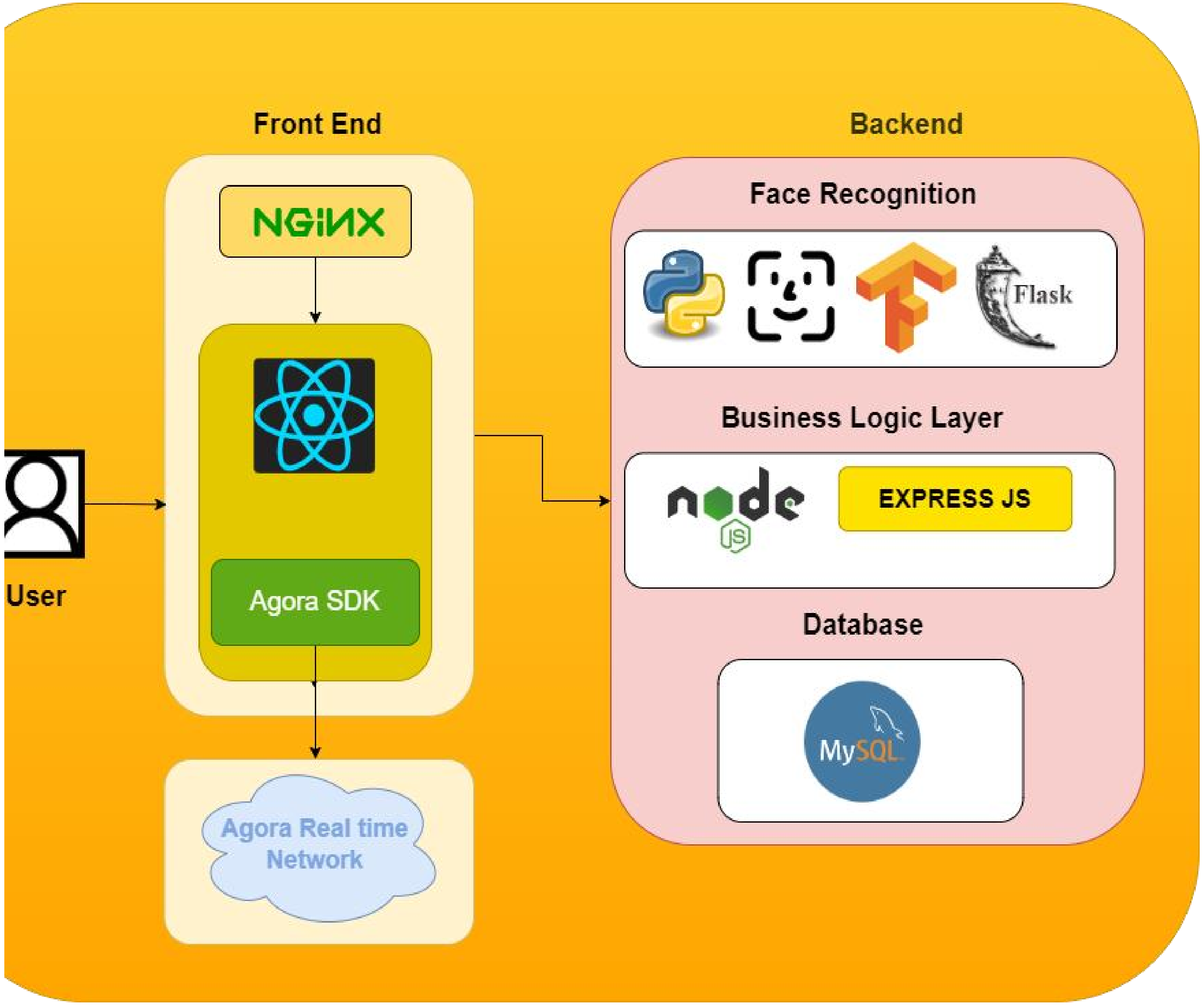
SIGN IN

Don't have an account yet? SIGN UP

Technologies

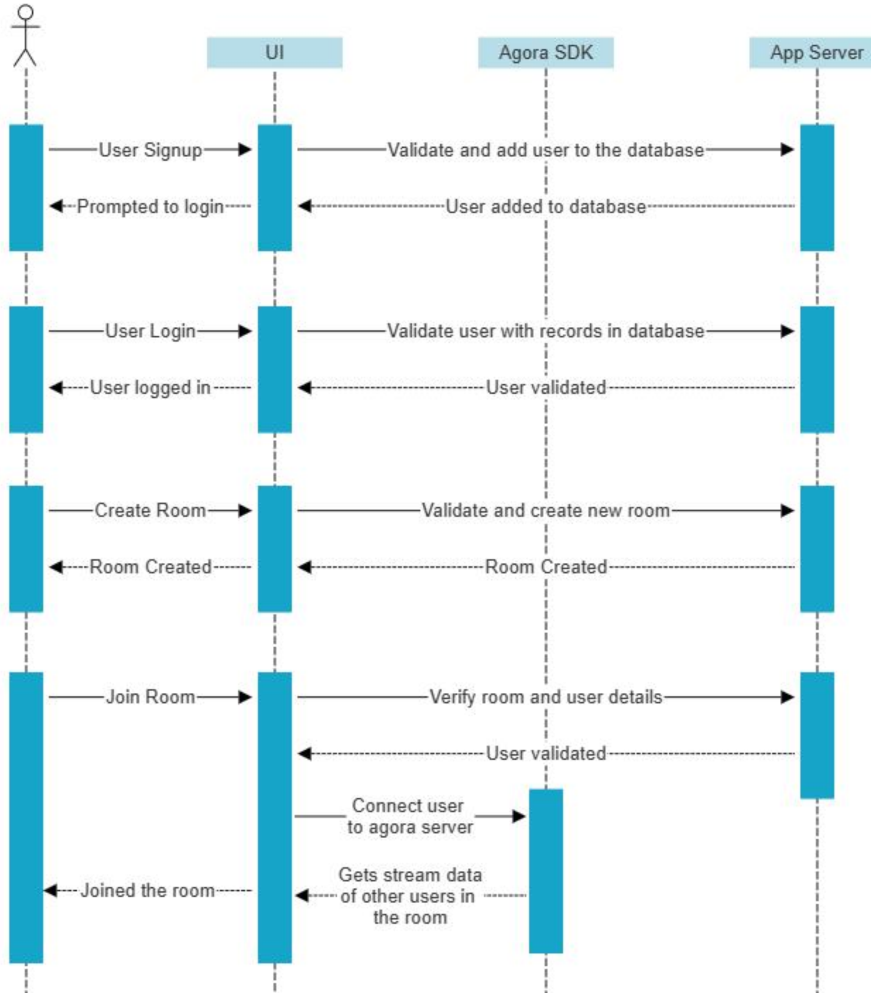


Architecture Diagram



changed

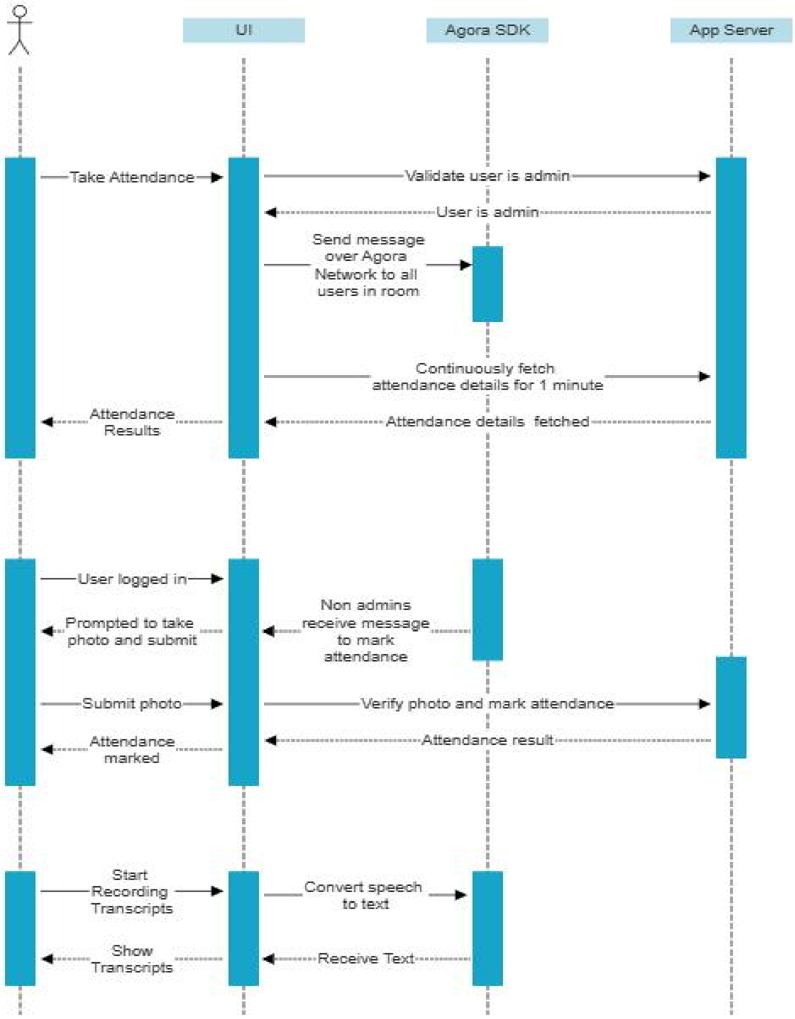
Sequence Diagram



UI
Agora SDK
App Server
User Signup
Validate and add user to the database
Prompted to login
User added to database
User Login
Validate user with records in database
User logged in
User validated
Create Room
Validate and create new room
Room Created
Room Created
Join Room
Verify room and user details
User validated
Connect user to agora server
Joined the room
Gets stream data of other users in the room

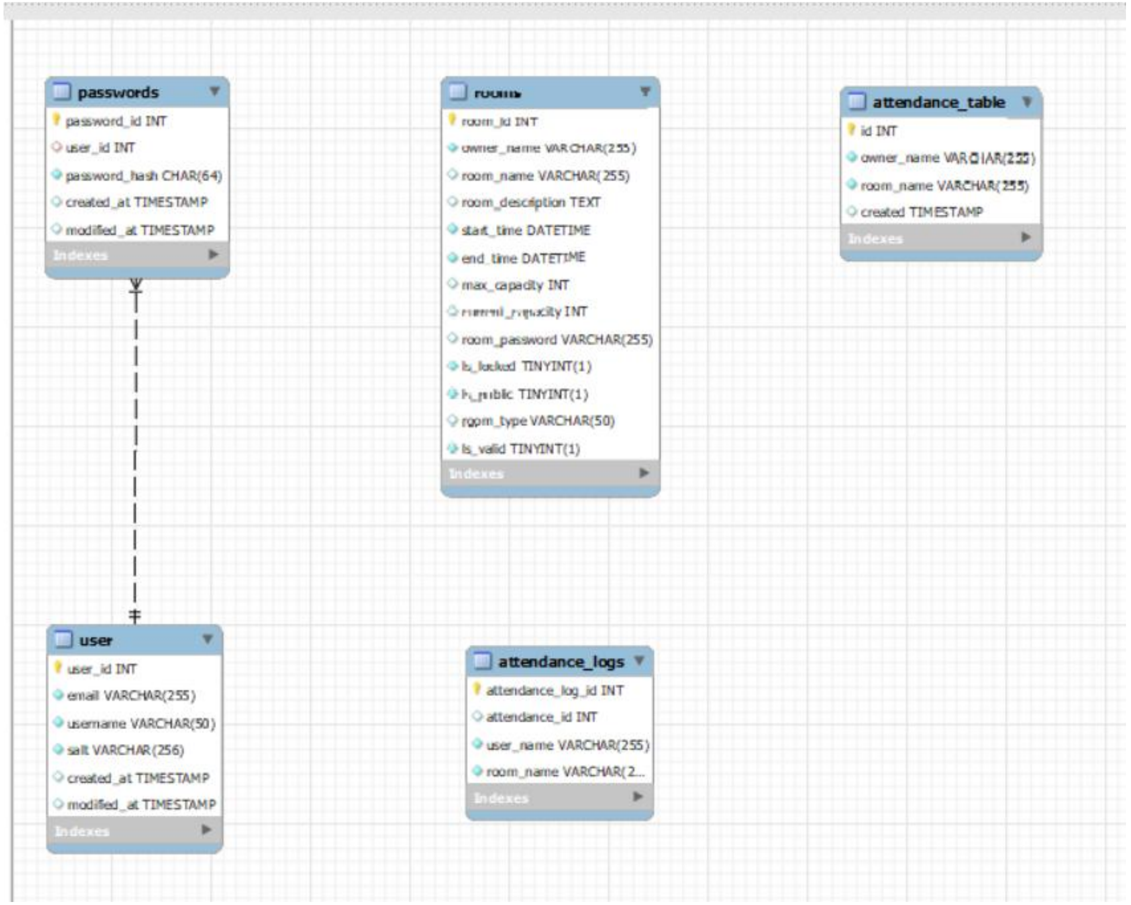
changed

Sequence Diagram


UI
Agora SDK
App Server
Take Attendance
Validate user is admin
User is admin
Send message over Agora Network to all users in room
Continuously fetch attendance details for 1 minute
Attendance Results
Attendance details fetched
User logged in
Prompted to take photo and submit
Non admins receive message to mark attendance
Submit photo
Verify photo and mark attendance
Attendance marked
Attendance result
Start Recording Transcripts
Convert speech to text
Show Transcripts
Receive Text

changed

ER Diagram



passwords
password_id INT
user_id INT
password_hash CHAR(64)
created_at TIMESTAMP
modified_at TIMESTAMP
Indexes

rooms
room_id INT
owner_name VARCHAR(255)
room_name VARCHAR(255)
room_description TEXT
start_time DATETIME
end_time DATETIME
max_capacity INT
current_capacity INT
room_password VARCHAR(255)
is_locked TINYINT(1)
is_public TINYINT(1)
room_type VARCHAR(50)
is_valid TINYINT(1)
Indexes

attendance_table
id INT
owner_name VARCHAR(255)
room_name VARCHAR(255)
created TIMESTAMP
Indexes

user
user_id INT
email VARCHAR(255)
username VARCHAR(50)
salt VARCHAR(256)
created_at TIMESTAMP
modified_at TIMESTAMP
Indexes

attendance_logs
attendance_log_id INT
attendance_id INT
user_name VARCHAR(255)
room_name VARCHAR(2...
Indexes

changed

SPRINT-4 RECAP

Enable screen sharing feature

Attendance automation using Face recognition

changed

PRODUCT BACKLOG & ACCEPTANCE CRITERIA

Userstory	Acceptance Criteria
Create Login and Signup pages	As a user Given user is on the login page. Then user should see username field, password field and login button.
	As a user Given user is on login page. When user clicks on the signup. Then sign up form should be visible.
	As a user Given user is on the signup page. Then user should see email field, username field, password field, verify password field and signup button.
Create Tables and stored procedures for authentication	As a user Given user is on the sign up page. When the user enters username And the user enters email And the user enters password And the user clicks on signup Then user should see the message "Registered Successfully"
	As a user Given user is on login page When user enters username And user enters password And user clicks on Login button Then user should be logged into the application

changed

PRODUCT BACKLOG & ACCEPTANCE CRITERIA

User Story	Acceptance Criteria
Create APIs for user signup and login	As a developer Given Host consumes APIs When Host provides proper input to the APIs Then API should validate user AND respond with proper response
Implement Backend authentication using JWT token	As a developer Given Host is consuming login API When Host calls API with email and password Then Backend should return with JWT token if user credentials are valid
Integration of authentication with existing application	As a user Given Host is logged in And Host is on Home page When user tries to access the application Then details should be fetched based on JWT token

changed

PRODUCT BACKLOG & ACCEPTANCE CRITERIA

User Story	Acceptance Criteria
Deploy whole application on a public server	As a user Given Host connected to internet When Host goes to URL https://simplyonline.com Then Host should access the simply online application
Group chat	As a user Given Host is in online room When Host clicks on chat icon Then Host should see group chat And Host should be able to send message to group
Transcripts	As a user Given Host is in online room When Host clicks on transcripts icon Then Host should see live captions
Screenshare feature	As an admin Given Host is in online room When Host clicks on screenshare icon Then Host should share his screen to whole room

changed

SPRINT 5 BACKLOG

STORY ID	USER STORY	STORY POINTS
SIM-42	Create login and signup pages	8
SIM-43	create tables and stored procedures for authentication	8
SIM-45	implement backend authentication using JWT token	13
SIM -44	create API's for user signup and login	13
SIM-47	testing login and signup features	13
SIM-46	integration of authentication with existing application	5

changed

TEST CASES

Unit to test	Scenario	Expected Result
Sign up user	User provides valid inputs	User will be signed up and will be redirected to login page
Sign up user	User provides invalid email format	Error should be shown to user that email is not valid
Signup user	User provides invalid password	Error should be shown to user that password is not valid or passwords does not match
Login	User provides valid credentials	User should be logged in and redirected to home page
Login	User provides invalid credentials	User should see an error message that credentials are not valid
API validations	Call APIs with wrong inputs	Same errors should be thrown from backend

changed

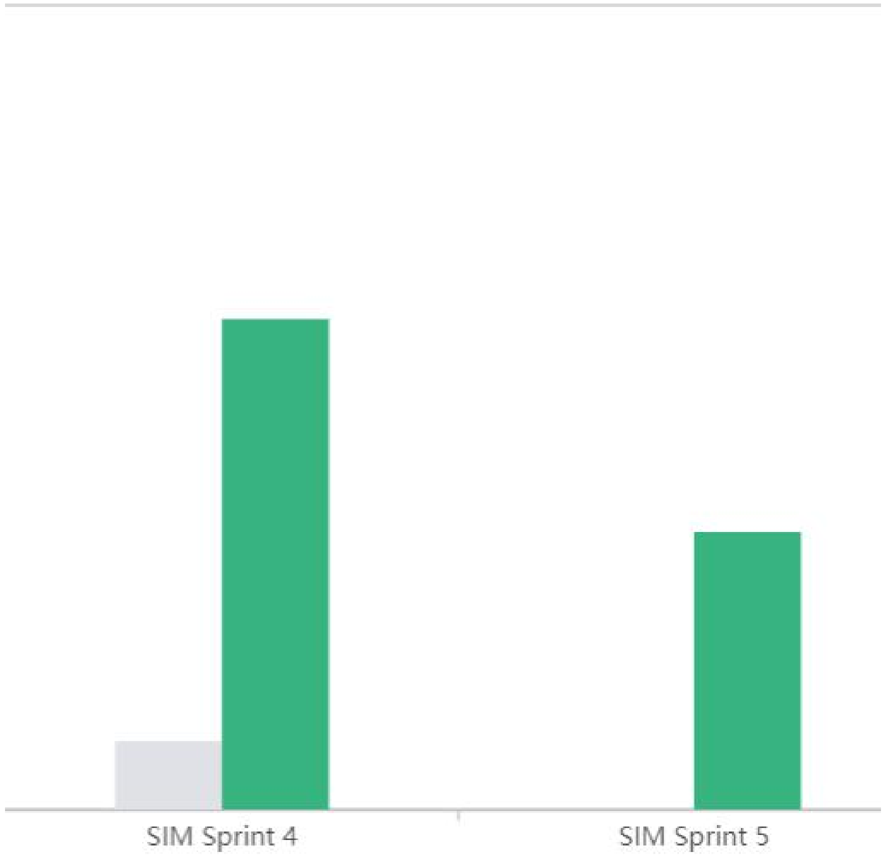
STORY ID	USER STORY	STORY POINTS	Status
SIM-42	Create login and signup pages	8	Done
SIM-43	create tables and stored procedures for authentication	8	Done
SIM-45	implement backend authentication using JWT token	13	Done
SIM -44	create API's for user signup and login	13	Done
SIM-47	testing login and signup features	13	Done
SIM-46	integration of authentication with existing application	5	Done

COMPLETED AND NOT COMPLETED TASKS

METRICS

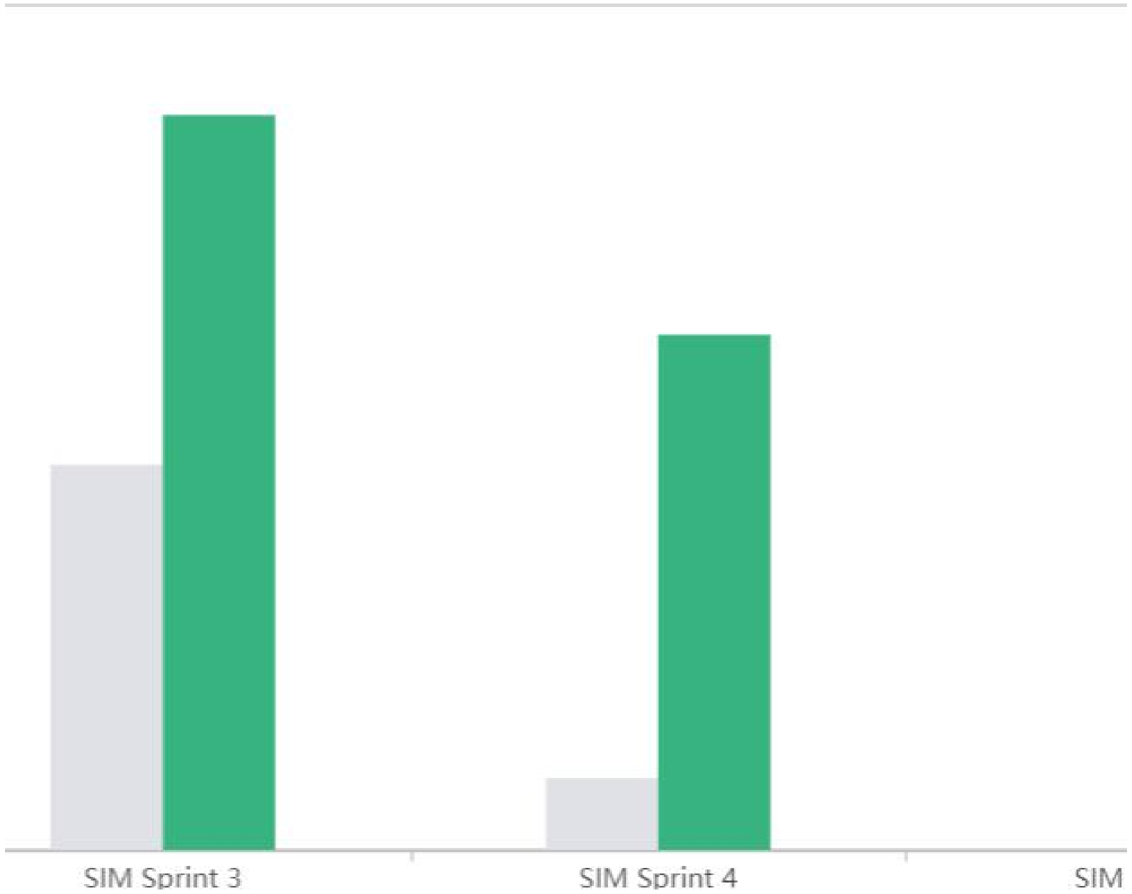
changed

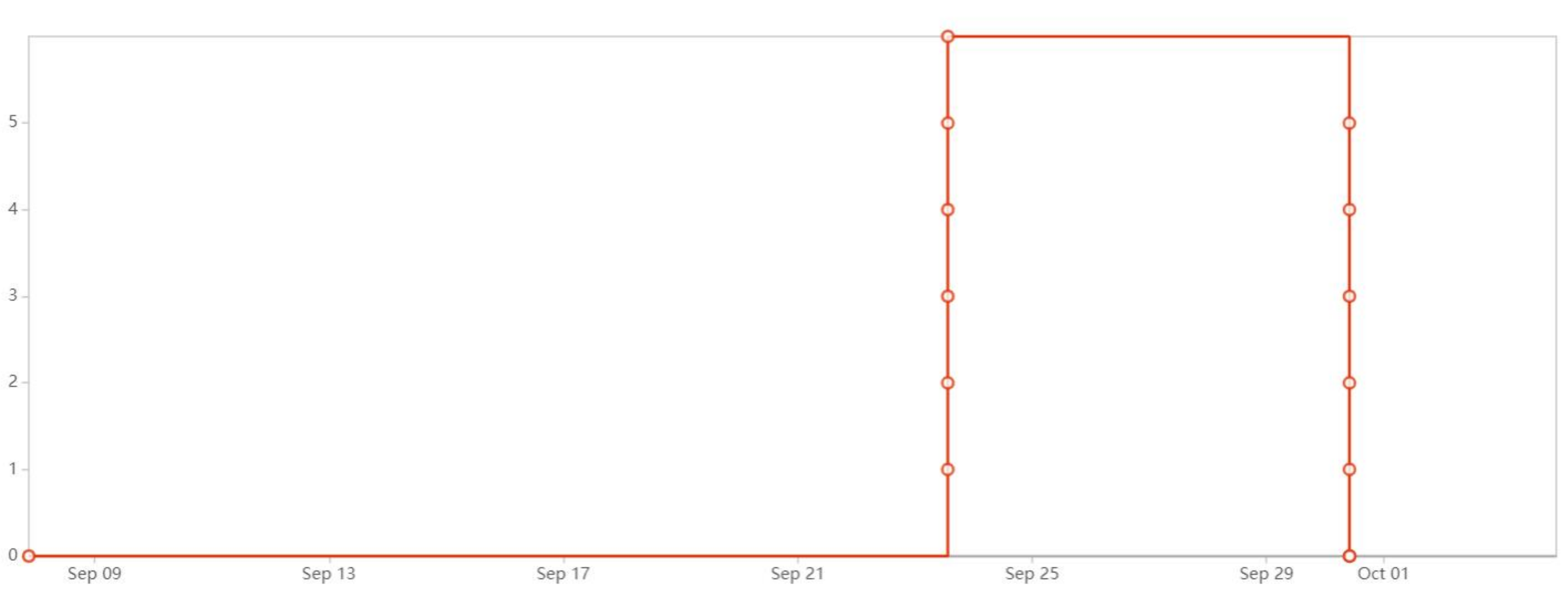
Team Velocity Report



changed

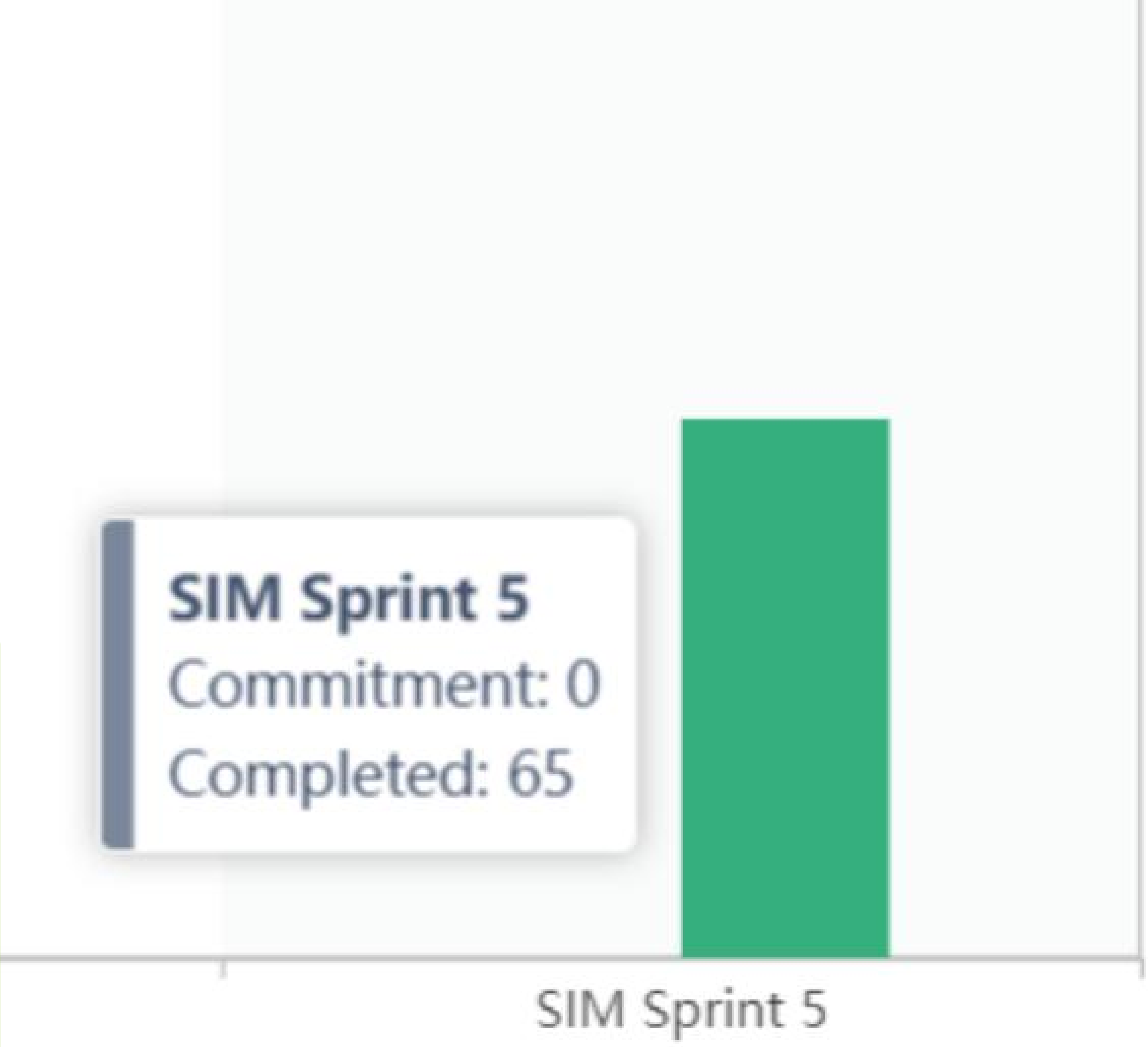
Team Historical Velocity Report





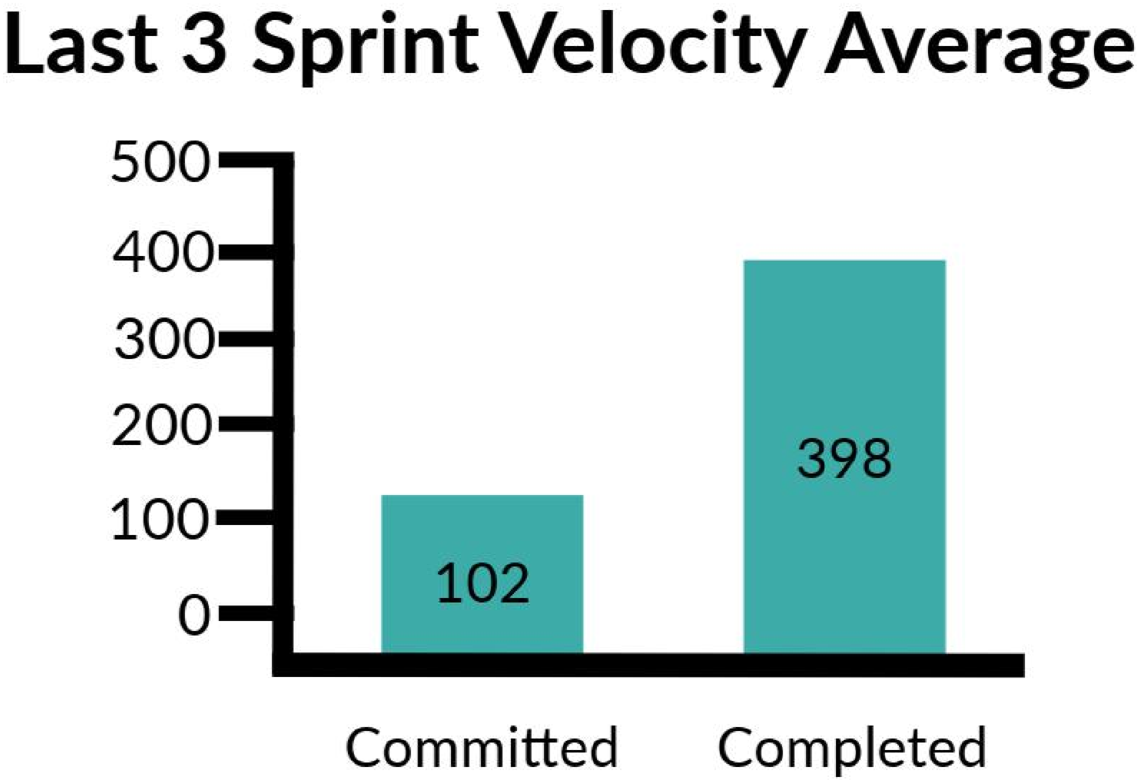
SPRINT BURNDOWN CHART

changed



Committed and Completed Ratio

Average Velocity



RETROSPECTIVE

- **What went well:**
- Team successfully completed all the planned user stories or tasks during sprint
- Team's ability to quickly identify and resolve issues, blockers, or impediments during the sprint

- **What didn’t go well:**
- Sprint planning didn't go well at the beginning and this made it difficult to allocate resources effectively.

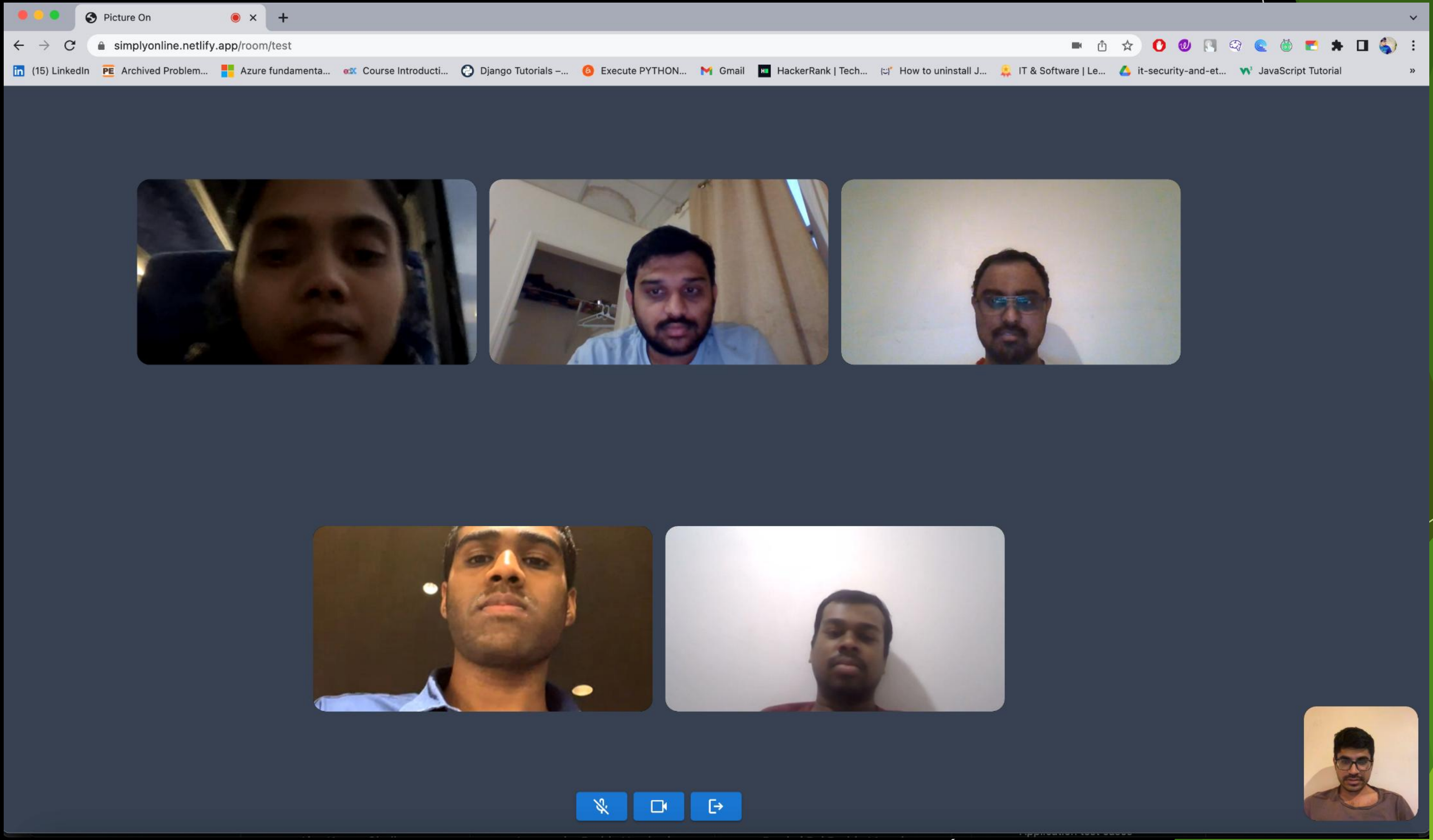
- **Action Steps:**
- Break down user stories and tasks into smaller, manageable sub-tasks.
- Clearly define sprint goals, tasks, and priorities. Conduct regular meetings to track the sprint progress

changed

SPRINT 6

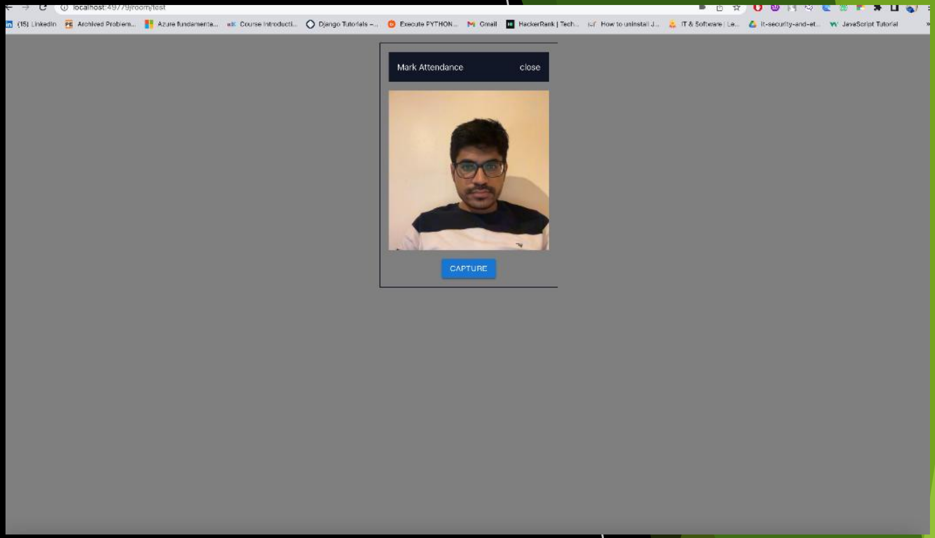
STORY ID	USER STORY	STORY POINTS
SIM–21	Enable screen sharing feature	20
SIM–39	Mark Attendance feature production deployment	40
SIM–40	Test live application with many users	20
SIM–49	Deploy application on public server	40
SIM–48	Add transcript functionality	30

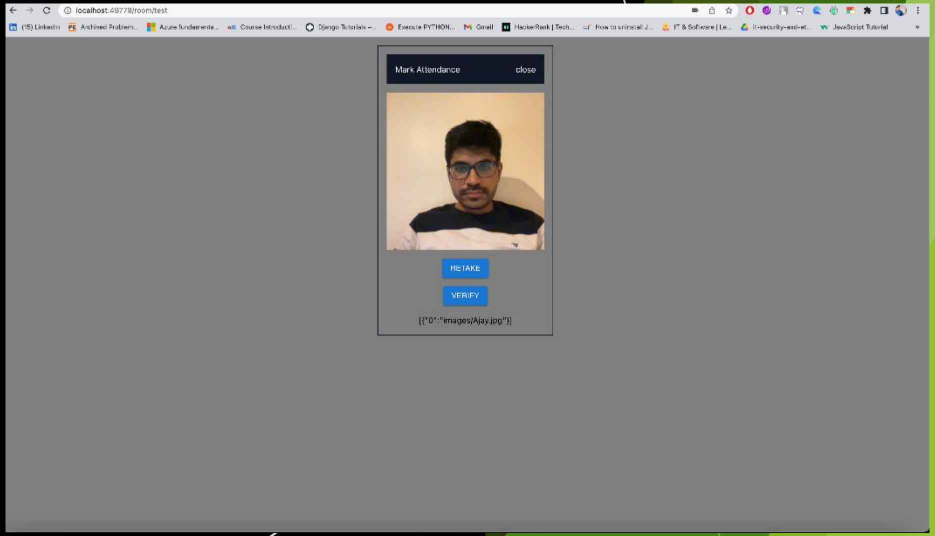
GROUP CALL


Picture On
simplyonline.netlify.app/room/test
(15) LinkedIn
Archived Problem...
Azure fundamenta...
Course Introducti...
Django Tutorials –...
Execute PYTHON...
Gmail
HackerRank | Tech...
How to uninstall J...
IT & Software | Le...
it-security-and-et...
JavaScript Tutorial

Mark Attendance/verify

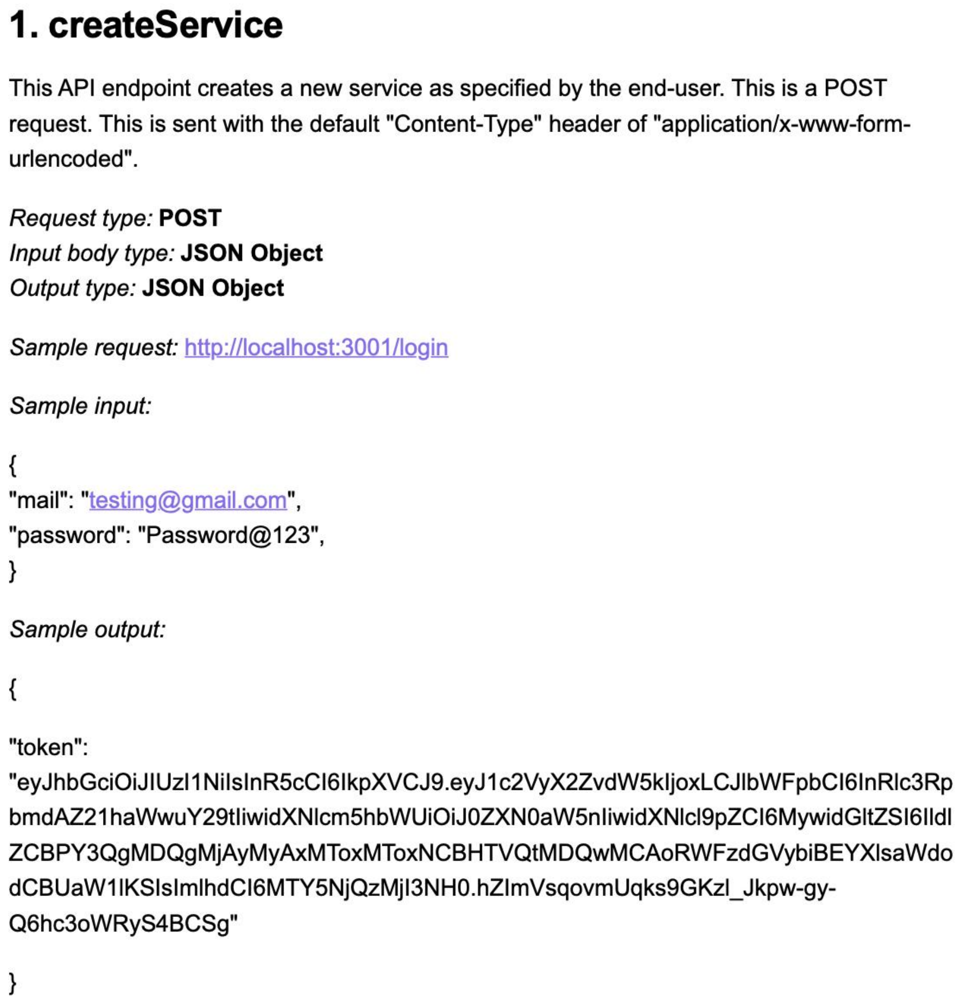
- This screen allows user to capture the photo and verify that it is the same user or not





```
[{"0":"images/Ajay.jpg"}]
```

API for login and signup



1. createService

This API endpoint creates a new service as specified by the end-user. This is a POST request. This is sent with the default "Content-Type" header of "application/x-www-form-urlencoded".

Request type: **POST**
Input body type: **JSON Object**
Output type: **JSON Object**

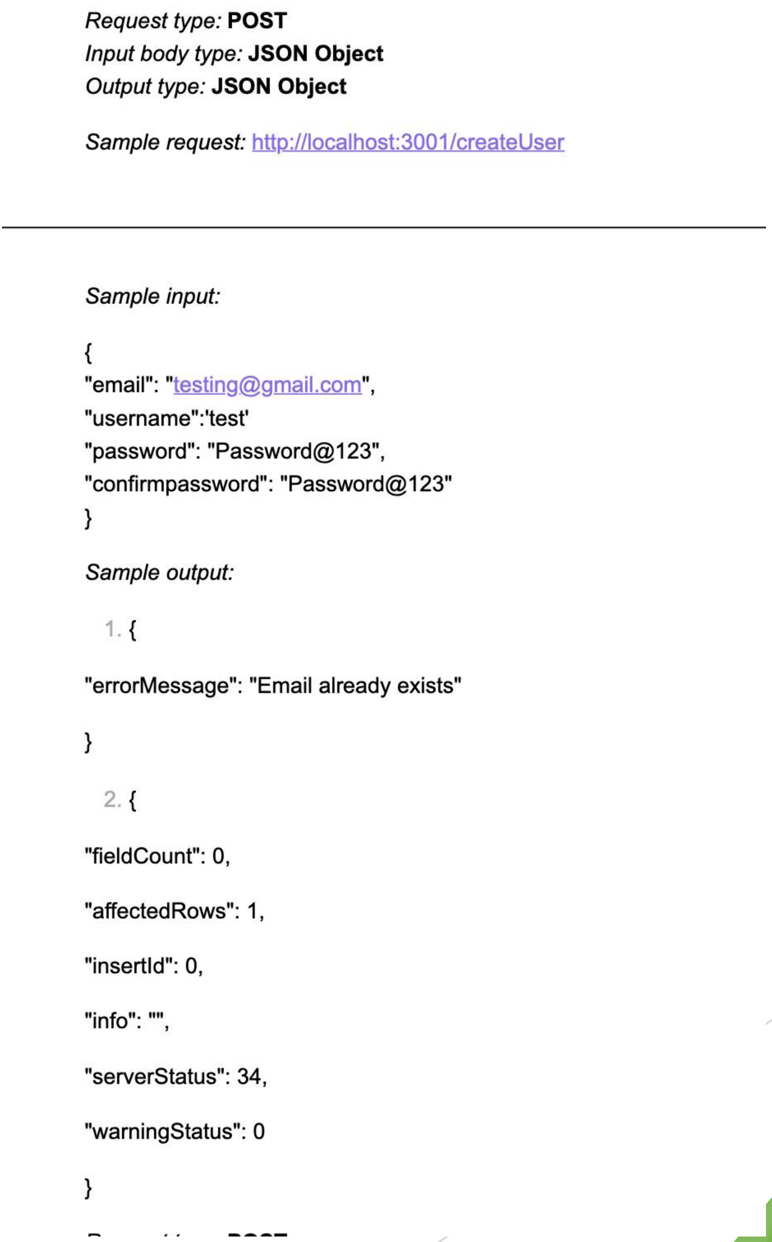
Sample request: http://localhost:3001/login

Sample input:

```
{
"mail": "testing@gmail.com",
"password": "Password@123",
}
```

Sample output:

```
{

"token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyX2ZvdW5kIjoxLCJlbWFpbCI6InRlc3Rp
bmdAZ21haWwuY29tIiwidXNlcm5hbWUiOiJ0ZXN0aW5nIiwidXNlcl9pZCI6MywidGltZSI6Ildl
ZCBPY3QgMDQgMjAyMyAxMToxMToxNCBHTVQtMDQwMCAoRWFzdGVybiBEYXlsaWdo
dCBUaW1lKSIsImlhdCI6MTY5NjQzMjI3NH0.hZImVsqovmUqks9GKzl_Jkpw-gy-
Q6hc3oWRyS4BCSg"

}
```



Request type: **POST**
Input body type: **JSON Object**
Output type: **JSON Object**

Sample request: http://localhost:3001/createUser

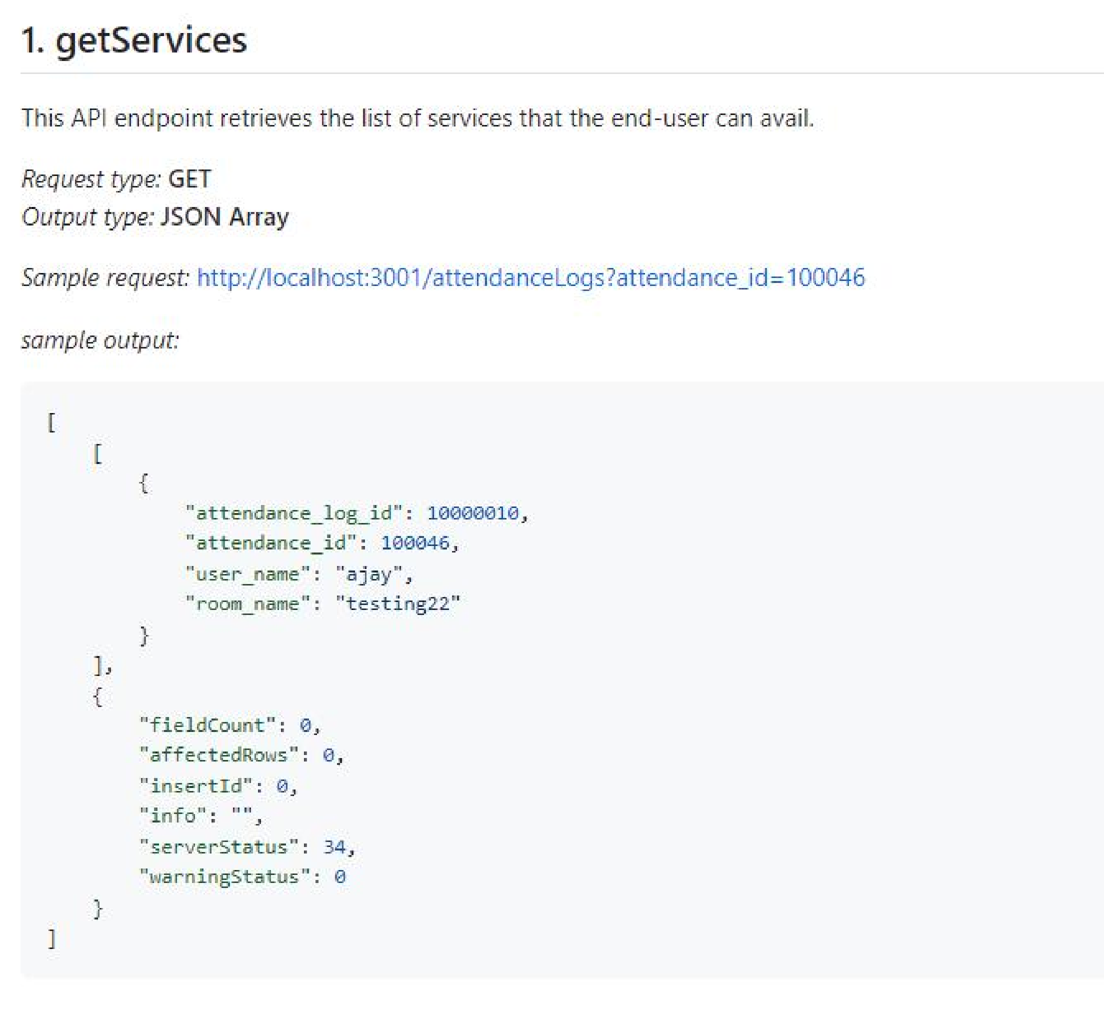
Sample input:

```
{
"email": "testing@gmail.com",
"username":'test'
"password": "Password@123",
"confirmpassword": "Password@123"
}
```

Sample output:

```
  1. {

"errorMessage": "Email already exists"

}

  2. {

"fieldCount": 0,

"affectedRows": 1,

"insertId": 0,

"info": "",

"serverStatus": 34,

"warningStatus": 0

}
```

Api’s



1. getServices

This API endpoint retrieves the list of services that the end-user can avail.

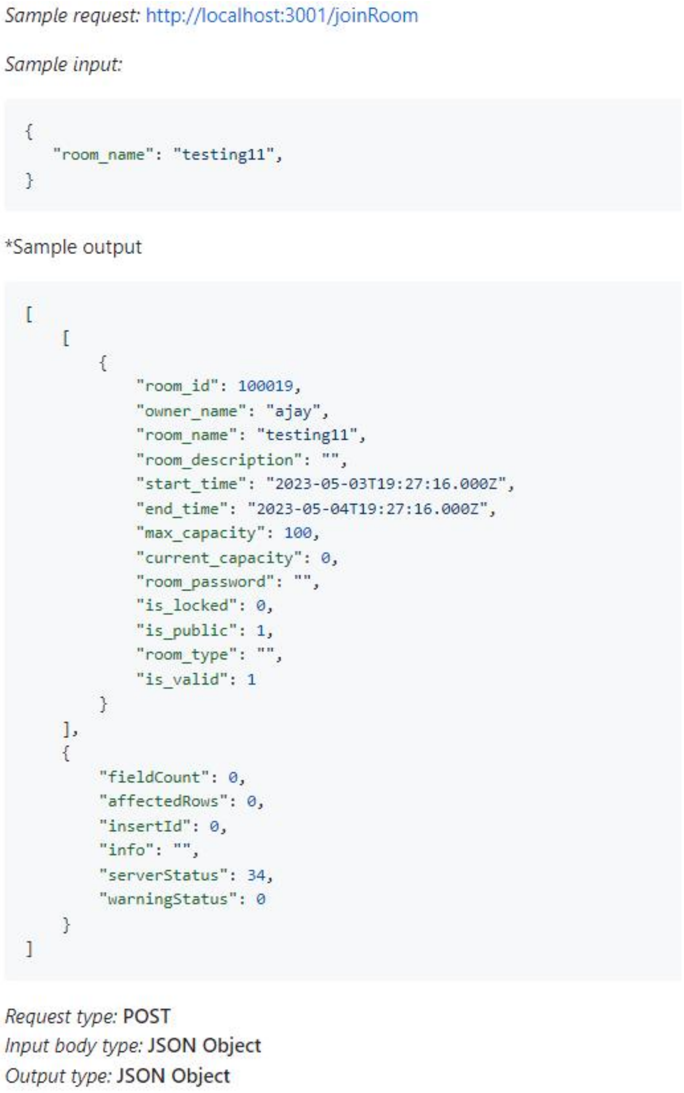
Request type: GET
Output type: JSON Array

Sample request: http://localhost:3001/attendanceLogs?attendance_id=100046

sample output:

```
[
    [
        {
            "attendance_log_id": 10000010,
            "attendance_id": 100046,
            "user_name": "ajay",
            "room_name": "testing22"
        }
    ],
    {
        "fieldCount": 0,
        "affectedRows": 0,
        "insertId": 0,
        "info": "",
        "serverStatus": 34,
        "warningStatus": 0
    }
]
```

API’s



Sample request: http://localhost:3001/joinRoom

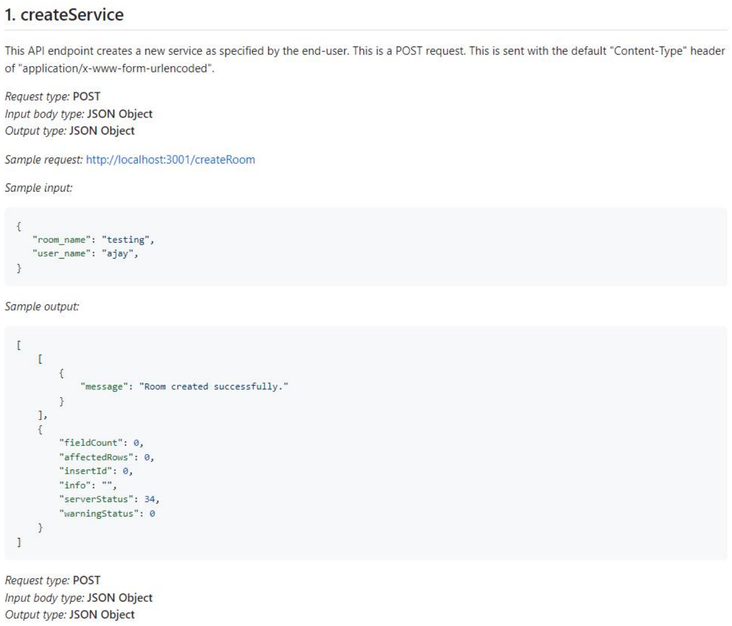
Sample input:

```
{
    "room_name": "testing11",
}
```

*Sample output

```
[
    [
        {
            "room_id": 100019,
            "owner_name": "ajay",
            "room_name": "testing11",
            "room_description": "",
            "start_time": "2023-05-03T19:27:16.000Z",
            "end_time": "2023-05-04T19:27:16.000Z",
            "max_capacity": 100,
            "current_capacity": 0,
            "room_password": "",
            "is_locked": 0,
            "is_public": 1,
            "room_type": "",
            "is_valid": 1
        }
    ],
    {
        "fieldCount": 0,
        "affectedRows": 0,
        "insertId": 0,
        "info": "",
        "serverStatus": 34,
        "warningStatus": 0
    }
]
```

Request type: POST
Input body type: **JSON Object**
Output type: **JSON Object**



1. createService

This API endpoint creates a new service as specified by the end-user. This is a POST request. This is sent with the default "Content-Type" header of "application/x-www-form-urlencoded".

Request type: POST
Input body type: JSON Object
Output type: JSON Object

Sample request: http://localhost:3001/createRoom

Sample input:

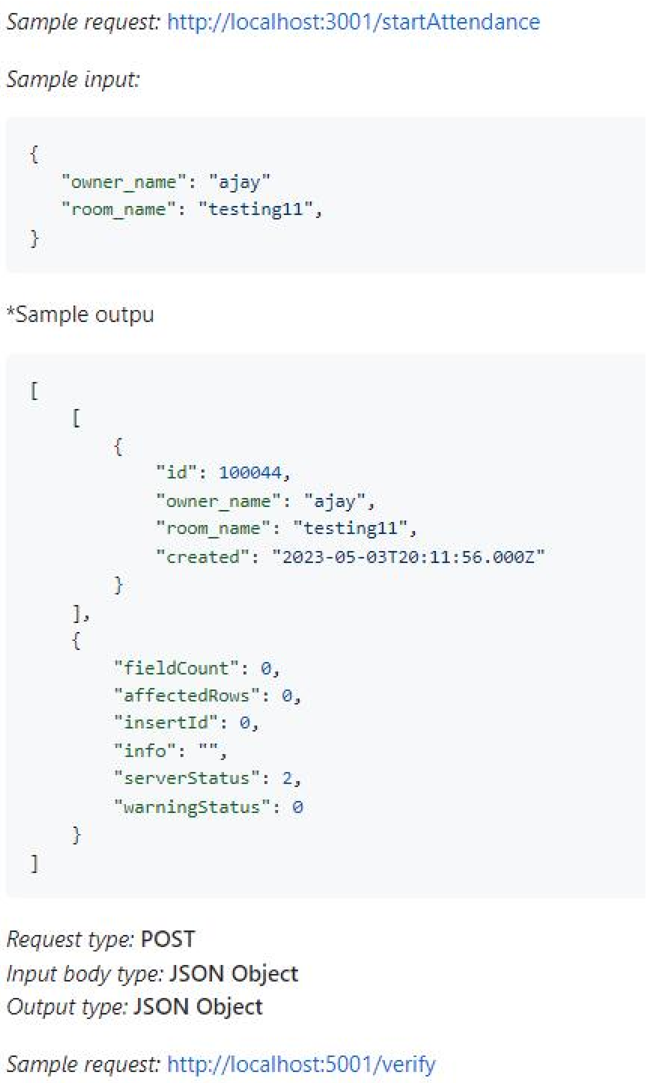
```
{
   "room_name": "testing",
   "user_name": "ajay",
}
```

Sample output:

```
[
    [
        {
            "message": "Room created successfully."
        }
    ],
    {
        "fieldCount": 0,
        "affectedRows": 0,
        "insertId": 0,
        "info": "",
        "serverStatus": 34,
        "warningStatus": 0
    }
]
```

Request type: POST
Input body type: JSON Object
Output type: JSON Object

Api’s



Sample request: http://localhost:3001/startAttendance

Sample input:

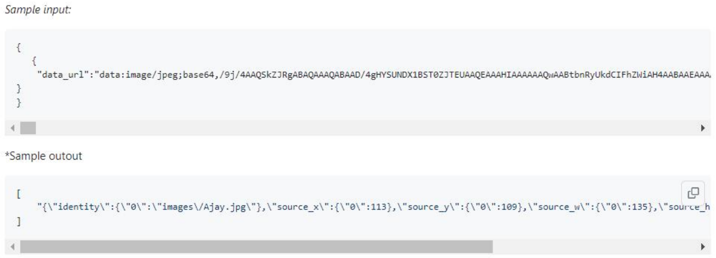
```
{
   "owner_name": "ajay"
   "room_name": "testing11",
}
```

*Sample outpu

```
[
    [
        {
            "id": 100044,
            "owner_name": "ajay",
            "room_name": "testing11",
            "created": "2023-05-03T20:11:56.000Z"
        }
    ],
    {
        "fieldCount": 0,
        "affectedRows": 0,
        "insertId": 0,
        "info": "",
        "serverStatus": 2,
        "warningStatus": 0
    }
]
```

Request type: POST
Input body type: JSON Object
Output type: JSON Object

Sample request: http://localhost:5001/verify



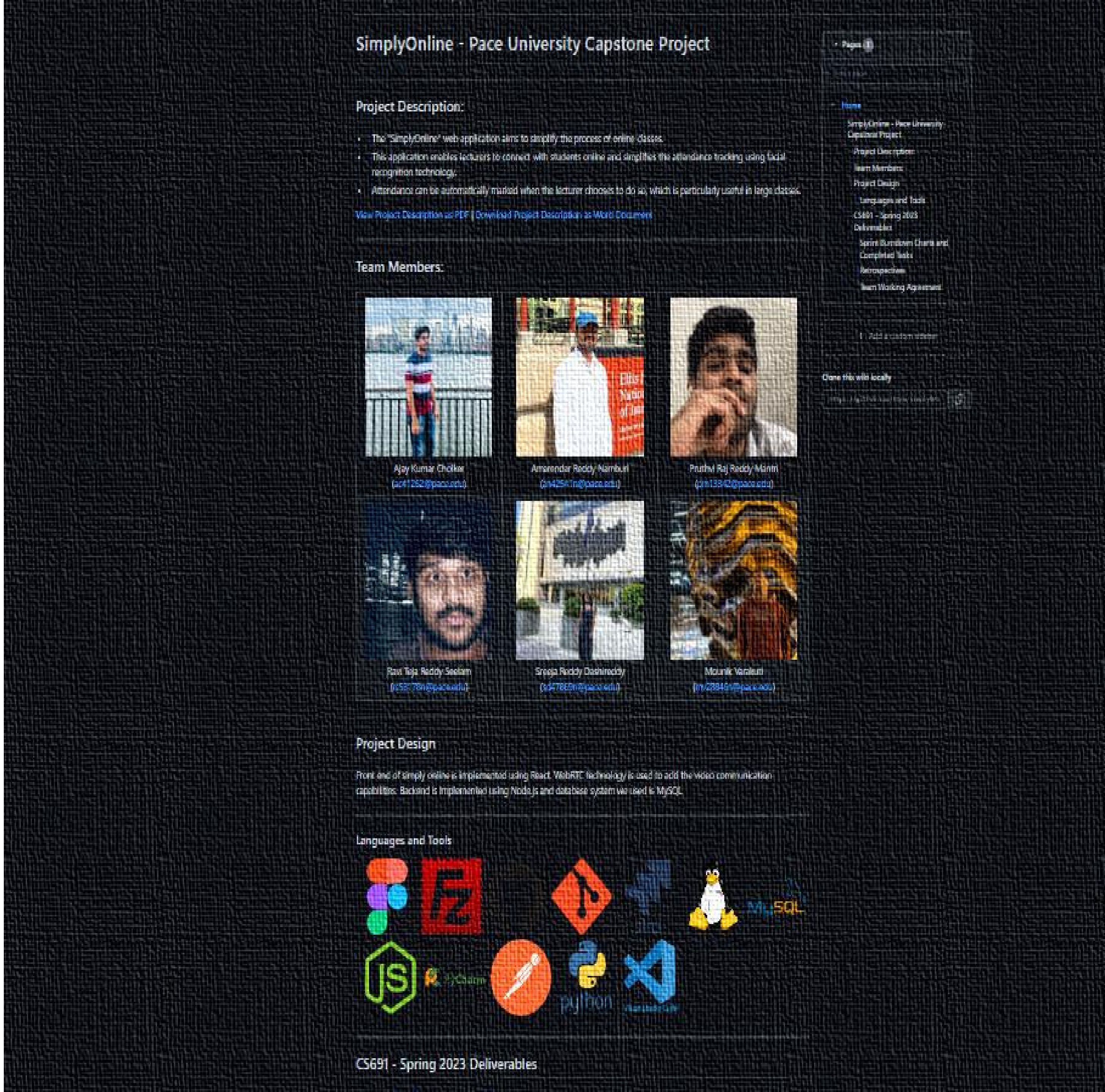
Sample input:

```
{
   {
    "data_url":"data:image/jpeg;base64,/9j/4AAQSkZJRgABAQAAAQABAAD/4gHYSUNDX1BST0ZJTEUAAQEAAAHIAAAAAAQwAABtbnRyUkdCIFhZWiAH4AABAAEAAA
}
}
```

*Sample outout

```
[
    "{\"identity\":{\"0\":\"images\/Ajay.jpg\"},\"source_x\":{\"0\":113},\"source_y\":{\"0\":109},\"source_w\":{\"0\":135},\"source_h
]
```

https://github.com/htmw/SimplyOnline/wiki



SimplyOnline - Pace University Capstone Project

Project Description:

- The "SimplyOnline" web application aims to simplify the process of online classes.
- This application enables lecturers to connect with students online and simplifies the attendance tracking using facial recognition technology.
- Attendance can be automatically marked when the lecturer chooses to do so, which is particularly useful in large classes.

View Project Description as PDF | Download Project Description as Word Document

Team Members:

Ajay Kumar Cholker (ac41262@pace.edu)

Amarendar Reddy Namburi (an42541n@pace.edu)

Pruthvi Raj Reddy Manni (pm13842@pace.edu)

Ravi Teja Reddy Seelam (rs53778n@pace.edu)

Sreeja Reddy Dashireddy (sd47869n@pace.edu)

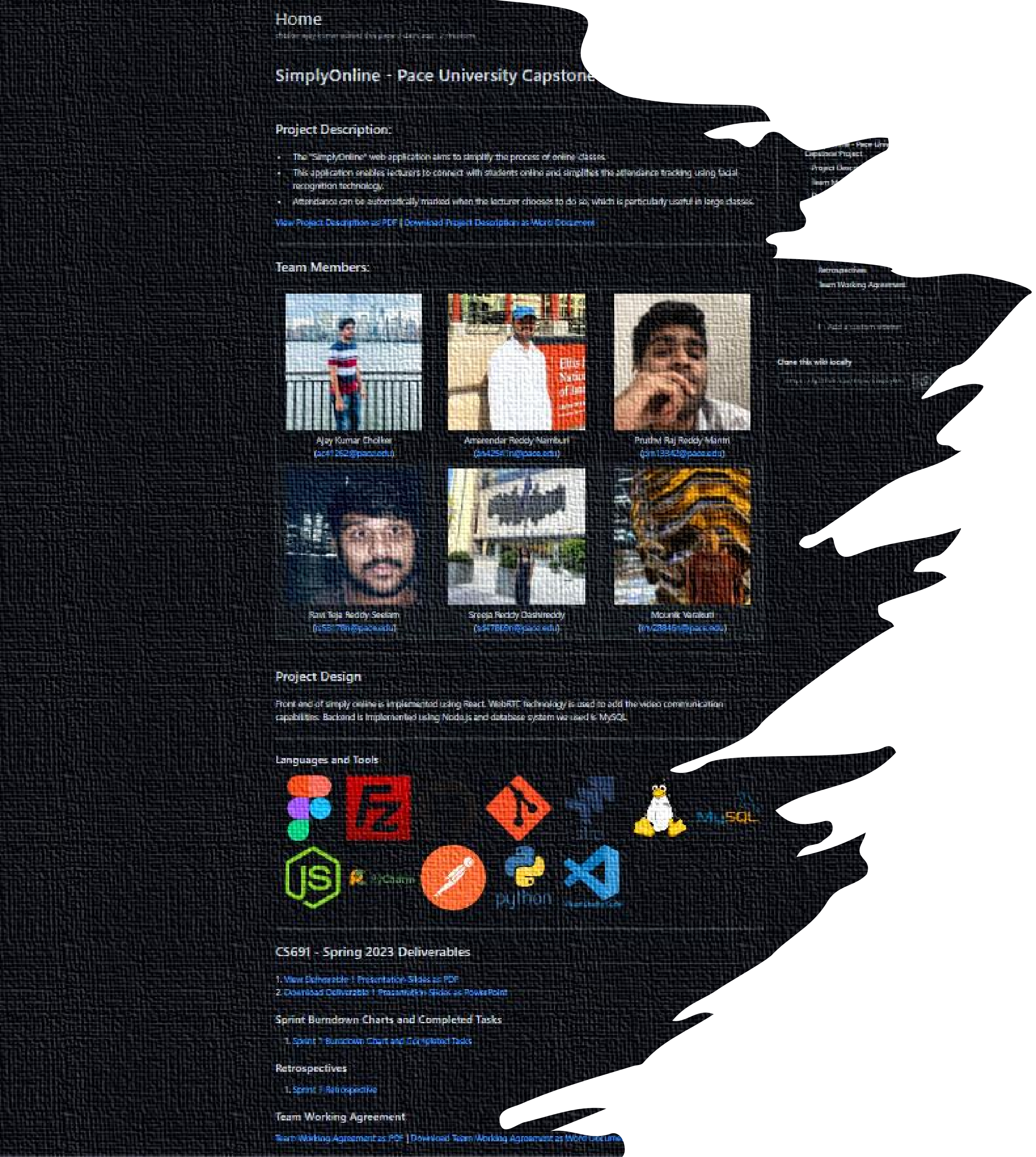
Mounik Varaluri (mv28846n@pace.edu)

Project Design

Front end of simply online is implemented using React. WebRTC technology is used to add the video communication capabilities. Backend is implemented using Node.js and database system we used is MySQL

Languages and Tools

CS691 - Spring 2023 Deliverables



Home

SimplyOnline - Pace University Capstone

Project Description:

- The "SimplyOnline" web application aims to simplify the process of online classes.
- This application enables lecturers to connect with students online and simplifies the attendance tracking using facial recognition technology.
- Attendance can be automatically marked when the lecturer chooses to do so, which is particularly useful in large classes.

View Project Description as PDF | Download Project Description as Word Document

Team Members:

Ajay Kumar Cholker (ac41262@pace.edu)

Amarendar Reddy Namburi (an42541n@pace.edu)

Pruthvi Raj Reddy Mantri (pm13342@pace.edu)

Ravi Teja Reddy Seelam (rs53178n@pace.edu)

Sreeja Reddy Dashireddy (sd47869n@pace.edu)

Mounik Varaluri (mv28846n@pace.edu)

Project Design

Front end of simply online is implemented using React. WebRTC technology is used to add the video communication capabilities. Backend is implemented using Node.js and database system we used is MySQL.

Languages and Tools

CS691 - Spring 2023 Deliverables

1. View Deliverable 1 Presentation Slides as PDF
2. Download Deliverable 1 Presentation Slides as PowerPoint

Sprint Burndown Charts and Completed Tasks

1. Sprint 1 Burndown Chart and Completed Tasks

Retrospectives

1. Sprint 1 Retrospective

Team Working Agreement

Team Working Agreement as PDF | Download Team Working Agreement as Word Docume

DEMO LINK

- YouTube

THANK YOU