Saravanakumar M

Internship report

WEB DEVELOPMENT-

Learn To Build a Real Time Website Like YouTube

**Introduction**

This project aimed to enhance a YouTube-inspired web application by integrating three key features: a points system based on user interactions with a custom video player, advanced gesture-based controls in the video player, and a VoIP feature for video calls, screen sharing, and recording. The goal was to create a more engaging and interactive user experience while ensuring ease of use and functionality.

**Background**

The web application is designed to offer users a comprehensive media experience, enabling them to create channels, upload videos, and view content similar to YouTube. The main tasks involved in the project included enhancing the app with a points system, a custom video player with gesture-based controls, and VoIP capabilities. These enhancements aimed to increase user engagement and provide additional functionalities that cater to modern digital consumption habits.

**Learning Objectives**

1. Understand the implementation of a points system based on user activity.
2. Integrate a custom video player with advanced gesture-based controls.
3. Implement a VoIP feature with screen sharing and recording capabilities.
4. Host the application on Netlify or Vercel.

**Activities and Tasks**

1. **Points System**
   * **Task:** Implement a system to allocate points for watching videos.
   * **Activity:** The task was attempted, but the implementation of the points system was not fully completed. A system was set up to track user interactions, but the full functionality and integration with user profiles were not achieved.
2. **Custom Video Player**
   * **Task:** Develop a custom video player with gesture-based controls.
   * **Activity:** Successfully completed. The video player was equipped with gesture-based controls, including:
     + Double-tap on the right side: Playback moves 10 seconds forward
     + Double-tap on the left side: Playback moves 10 seconds backward
     + Single-tap in the middle: Video paused
     + Triple-tap in the middle: Move to the next video
     + Triple-tap on the right side: Close the website
     + Triple-tap on the left side: Show the comment section
     + Single-tap on the top right corner: Display current location and temperature
     + Hold the right side: Video speed two times faster
     + Hold the left side: Video speed two times slower
3. **VoIP Feature**
   * **Task:** Add video calling capabilities with screen sharing and recording.
   * **Activity:** About 50% completed with some errors. The VoIP feature was partially implemented, including basic video calling and screen sharing functions. However, some errors and incomplete functionalities limited the full effectiveness of this feature.

**Skills and Competencies**

* Proficiency in React for building interactive user interfaces.
* Experience with video player libraries and custom gesture controls.
* Knowledge of WebRTC or similar technologies for implementing VoIP features.
* Understanding of hosting web applications on platforms like Netlify or Vercel.

**Feedback and Evidence**

Feedback was collected through user testing sessions, focusing on the usability and responsiveness of the new features. Users appreciated the intuitive gesture controls and the partial functionality of the VoIP feature, though the incomplete points system and some issues with the VoIP feature were noted.

**Challenges and Solutions**

1. **Implementing Gesture Controls**
   * **Challenge:** Developing gesture controls while avoiding conflicts with default video behaviors.
   * **Solution:** Implemented gesture controls using references to the video element and handling different tap types with conditional statements.
2. **VoIP Feature**
   * **Challenge:** Implementing WebRTC and handling recording issues.
   * **Solution:** Used React Media Recorder for comprehensive recording of meetings, addressing issues with recording empty files.
3. **Implementing Video Stream on Vercel**
   * **Challenge:** User’s own stream was not visible when deployed on Vercel.
   * **Solution:** Used useEffect hooks to ensure proper rendering of the stream.
4. **Hosting and Deployment**
   * **Challenge:** Deployment issues with Vercel not supporting static file serving for the backend.
   * **Solution:** Deployed the backend on Render and the frontend on Vercel.

**Outcomes and Impact**

The project has made significant strides in enhancing the web application, though not all tasks were completed:

1. **Points System:**
   * **Status:** Attempted but not fully completed.
   * **Impact:** The incomplete points system limited its potential to increase user engagement through a reward-based mechanism.
2. **Custom Video Player:**
   * **Status:** Successfully completed.
   * **Impact:** The custom video player with gesture-based controls enhanced user interaction and experience with video content.
3. **VoIP Feature:**
   * **­Status:** About 50% completed with some errors.
   * **Impact:** The partial VoIP feature introduced some functionality but did not fully meet its potential, affecting its ability to enhance communication and collaboration.

**Conclusion**

The project successfully introduced a custom video player and made progress on the VoIP feature, though the points system and VoIP feature remain incomplete. The project demonstrates the potential for further development and refinement to fully achieve its goals and improve user engagement and functionality.