comp90082-2024-na-koala . . . 3
Team . . . 5
Project Brief . . . 6
ROSAnnotator: A Web Application for ROSBag Data Analysis in . . . 7
Project Overview, Background and Goal . . . 8
Project Specification with detailed Sprint planning . . . 9
Requirements & Design . . . 11
Motivational model . . . 15
DO-BE-FEEL . . . 16
Personas . . . 17
User Persona 1 . . . 19
User Persona 2 . . . 22
User Persona 3 . . . 25
User Persona 4 . . . 28
User Stories . . . 31
User Story Table . . . 33
Use Cases . . . 37
ROS Data Handling . . . 39
Dashboard Visualisation . . . 40
Synchronised Interaction . . . 41
Annotation Features . . . 42
Auto-Transcription . . . 44
Multiple ROSBags Loading . . . 45
Prototype . . . 46
Additional Helper Page . . . 47
Development . . . 48
Architecture Diagram . . . 49
Workflow Guide . . . 50
Frontend Documentation . . . 52
Development Environment . . . 53
Frontend Coding Standards . . . 56
Backend Documentation . . . 57
Deployment Instructions . . . 58
API Documentation . . . 60
Function Documentation . . . 64
Backend Coding Standards . . . 66
Technologies Used . . . 67
Testing Documentation . . . 68
Testing document Sprint3 . . . 71
Code Review . . . 79
2024-04-22 Code Review (Peer Review) . . . 80
2024-05-01 Code Review (Frontend ChatGPT Review) . . . 81
2024-05-02 Code Review (Backend ChatGPT Review) . . . 83
2024-05-11 Code Review . . . 85
2024-05-16 Code Review . . . 87

2024-05-17 Code Review . . . 89
2024-05-20 Code Review . . . 91
2024-05-21 Code Review . . . 93
Decision Making . . . 95
Decision log . . . 96
Decision on Speech-To-Text problem . . . 97
Decision on Predefined Booklist . . . 99
Decision on intro page . . . 101
Sprint Management with Trello Board . . . 102
Sprint 1 . . . 105
Sprint 1 Planning: Inception . . . 106
Sprint 1 check-list . . . 107
Sprint 1 task distribution . . . 109
Sprint 1 Retrospective Summary . . . 110
Sprint 2 . . . 111
Sprint 2 Planning: Development . . . 112
Sprint 2 checklist . . . 114
Sprint 2 Retrospective . . . 116
Sprint 3 . . . 118
Sprint 3 Planning: Development . . . 119
Sprint Velocity Estimation . . . 120
Sprint 3 Retrospective . . . 124
Sprint 3 Checklist . . . 126
Sprint 4 . . . 128
Sprint 4 Planning: Project Finalization and Delivery . . . 129
Sprint 4 Checklist . . . 130
Sprint 4 Retrospective . . . 131
Reports and Presentation Slide . . . 133
Cybersecurity Considerations . . . 134
Ethical Considerations . . . 136
Presentation Preparation . . . 138

comp90082-2024-na-koala

Team

Team

Owned by yucpeng1 • Updated on Mar 22, 2024

Name Student ID Email Position Responsibility Bowen Fan 1035162 bffa@student.unimelb.edu.au Product Manager/Frontend Developer Leads project direction and user interface development, ensuring alignment with user needs and seamless backend integration. Tianqi Wang 1045939 tww2@student.unimelb.edu.au Scrum Master/ROS Analyst Facilitates Agile process

Confluence

Open preview

Project Brief

Project Brief

Owned by yucpeng1 • Updated on Apr 27, 2024

🥸 Client Requirement https://chri-lab.notion.site/ROSAnnotator-A-Web-Application-for-ROSBag-Data-Analysis-in-Human-Robot-Interaction-e856d9fb941f4ceca0b4c53a77e71d03 📔 Overview & Background https://comp90082-2024-na-koala.atlassian.net/wiki/spaces/comp900822/pages/3735565 🎯 Requirement Analysis https://comp90082-2024-na-…

Confluence

Open preview

Requirements & Design

Requirements & Design

Owned by yucpeng1 • Updated on May 1, 2024

🥸 Client Requirements https://chri-lab.notion.site/ROSAnnotator-A-Web-Application-for-ROSBag-Data-Analysis-in-Human-Robot-Interaction-e856d9fb941f4ceca0b4c53a77e71d03 \uD83D\uDDD2 Project Overview The ROSAnnotator project aims to develop a standalone web application specifically designed to enhance the analysis of Robot Operating System Bag (ROSBag

Confluence

Open preview

Sprint Goals

Sprint Management with Trello Board

Owned by yucpeng1 • Updated on May 1, 2024

Sprint Task Tracking (Trello Board) Note: For each task breakdown, please read the ID as: “Epic Id . UId . Taks Id” (where UId is the user story Id. Please refer to the User Story Table) The Epic assignment is not displayed properly in the Confluence embedded link, please directly visit our Trello board to view the Epic assignment . https://trello

Confluence

Open preview

Meeting Notes

Meeting Notes

TW Owned by Tianqi Wang • Updated on Mar 19, 2024

Meeting Notes Incomplete Tasks Decisions

Confluence Open preview

Stand Up Meetings

Stand Up Meetings

Y Owned by yucpeng1 • Updated on Mar 19, 2024

Daily Stand Up

Confluence Open preview

Trello board

Sprint Management with Trello Board

Y Owned by yucpeng1 • Updated on May 1, 2024

Sprint Task Tracking (Trello Board) Note: For each task breakdown, please read the ID as: “Epic Id . UId . Taks Id” (where UId is the user story Id. Please refer to the User Story Table) The Epic assignment is not displayed properly in the Confluence embedded link, please directly visit our Trello board to view the Epic assignment . https://trello

Confluence Open preview

🦾 Project Brief

🤓 Client Requirement

ROSAnnotator: A Web Application for ROSBag Data Analysis in Human-Robot Interaction | Notion

Created by Wafa

Project Description

CHRI on Notion

Notion

Overview & Background

Project Overview, Background and Goal

Owned by Tianqi Wang • Updated on Mar 19, 2024

Project spec: https://chri-lab.notion.site/ROSAnnotator-A-Web-Application-for-ROSBag-Data-Analysis-in-Human-Robot-Interaction-e856d9fb941f4ceca0b4c53a77e71d03 Project Overview The ROSAnnotator project aims to develop a standalone web application specifically designed to enhance the analysis of Robot Operating System Bag (ROSBag) data, with a partic

Confluence

Open preview

🎯 Requirement Analysis

Requirements & Design

Owned by yucpeng1 • Updated on May 1, 2024

🤓 Client Requirements https://chri-lab.notion.site/ROSAnnotator-A-Web-Application-for-ROSBag-Data-Analysis-in-Human-Robot-Interaction-e856d9fb941f4ceca0b4c53a77e71d03 \uD83D\uDDD2 Project Overview The ROSAnnotator project aims to develop a standalone web application specifically designed to enhance the analysis of Robot Operating System Bag (ROSBag

Confluence

Open preview

📈 Specification & Planning

Project Specification with detailed Sprint planning

Owned by Tianqi Wang • Updated on Apr 27, 2024

The project spec for ROSAnnotator outlines a comprehensive plan to develop a standalone web application for the analysis of ROSBag data, specifically focusing on Human-Robot Interaction (HRI). The objectives and methodology indicate a multi-phase approach to the project, emphasizing data handling, user interaction, and innovative features for annot

Confluence

Open preview

ROSAnnotator: A Web Application for ROSBag Data Analysis in

https://chri-lab.notion.site/ROSAnnotator-A-Web-Application-for-ROSBag-Data-Analysis-in-Human-Robot-Interaction-e856d9fb941f4ceca0b4c53a77e71d03

Project Overview, Background and Goal

Project spec: ROSAnnotator: A Web Application for ROSBag Data Analysis in Human-Robot Interaction | Notion

Project Overview

The ROSAnnotator project aims to develop a standalone web application specifically designed to enhance the analysis of Robot Operating System Bag (ROSBag) data, with a particular emphasis on Human-Robot Interaction (HRI). ROSBags are a crucial element in robotics research, serving as a standard format for logging and replaying messages within the ROS ecosystem. These logs can include a wide variety of data types, such as video streams, 3D point clouds, and custom messages tailored for HRI studies. The application is envisioned to be a versatile tool for researchers, enabling the loading of multiple ROSBags, integration with auto-transcription tools for audio data processing, and the provision of a synchronized, interactive dashboard for intuitive data visualization.

Background

Human-Robot Interaction (HRI) is a growing field of study that explores how humans interact with robots in various contexts, from industrial applications to personal assistance and beyond. The analysis of HRI data is complex, involving multiple modalities such as visual data, audio communications, and sensor data from the robot. The ROS ecosystem provides a flexible framework for robot development and research, but the analysis of ROSBag data, especially from HRI experiments, requires specialized tools. Existing tools like Elan offer some capabilities for annotation and analysis but may not fully meet the unique needs of HRI research, such as handling specific ROS data types or synchronizing multiple data streams.

Goal

The primary goal of ROSAnnotator is to fill this gap by providing a comprehensive tool that facilitates the detailed analysis of HRI experiments. The project focuses on several key objectives:

- **Data Handling Capabilities:** To build robust support for reading and parsing the diverse data types contained in ROSBags, ensuring researchers can access and analyze all relevant data components of their HRI experiments.
- **Synchronised Interactive Dashboard:** To develop an intuitive, user-friendly interface that displays various data streams in a synchronized manner, allowing researchers to interact with and analyze data more effectively.
- **Annotation Features:** To enable detailed annotations of HRI interactions, including custom scales for states (e.g., "child looking at the robot") and events (e.g., "child entering input on the table"), thereby enhancing the depth and specificity of analysis.
- **Integration and Automation:** Although given less priority initially, integrating with auto-transcription tools for audio data and exploring automatic annotation methods based on audio transcripts are also envisioned to streamline the annotation process and make the tool more versatile.

Project Specification with detailed Sprint planning

The project spec for ROSAnnotator outlines a comprehensive plan to develop a standalone web application for the analysis of ROSBag data, specifically focusing on Human-Robot Interaction (HRI). The objectives and methodology indicate a multi-phase approach to the project, emphasizing data handling, user interaction, and innovative features for annotation and analysis.

Specific Requirements & Breakdown

Overall Project Requirements (for reference and future sprints)

1. **Data Handling Capabilities**
   - Develop parsing capabilities for video streams, 3D point clouds, and `hri_msgs`.
2. **Multiple ROSBags Loading**
   - Implement a feature to manage and analyze multiple ROSBags.
3. **Integration with Auto-Transcription**
   - Connect with an auto-transcription tool for audio to text conversion.
4. **Synchronised Interactive Dashboard**
   - Design and implement a user interface for data visualization.
5. **Annotation Features**
   - Develop custom scales and annotation tools for users.
6. **Automatic Annotation Exploration**
   - Research and prototype automatic annotation methods.

Please navigate to a detailed requirement specification and distribution using the below table.

Sprint 1	**Sprint 1 Planning: Inception** HW Owned by Harry Wang • Updated on Apr 27, 2024 Sprint 1: Requirement Analysis and Planning Phase Gather Detailed Requirements and Specifications Meetings with PhD students for requirements gathering. Identifying user stories based on these requirements. Survey of Existing Annotation Tools Research and document features of existing tools (e.… Confluence Open preview
Sprint 2	**Sprint 2 Planning: Development** YZ Owned by Yujie Zheng • Updated on Apr 30, 2024 Sprint 2 Planning: Development (March 25 - April 26) Overview Sprint 2 will span from March 25 to April 26, focusing on enhancing the core functionality and user interface of our audio-to-transcript application with speaker separation. This period aims to solidify our backend infrastructure, streamline front-end… Confluence Open preview
Sprint 3	**Sprint 3 Planning: Development** YZ Owned by Yujie Zheng • Updated on May 23, 2024 Sprint 3 Planning: Development（April 27 - May 24) Objective: Focus on refinement, and addressing any identified issues or improvements. Duration: 4 weeks Integration with Auto-Transcription Integrate auto-transcription service for speech-to-text conversion. Develop annotation functionalities. Duration: 1 Week… Confluence Open preview
Sprint 4	

Sprint Planning Notes

- **Sprint Duration**
  - Inception: 2 weeks
  - Development: 4 weeks
  - Handover: 2 weeks
- **Daily Standups** to discuss progress, obstacles, and next steps.
- **Review Meeting** at the end of Sprint 1 to evaluate the deliverables and plan for Sprint 2, focusing on System Design.
- **Retrospective Meeting** to discuss what went well, what didn’t, and how processes can be improved moving forward.

🎯 Requirements & Design

🤓 Client Requirements

ROSAnnotator: A Web Application for ROSBag Data Analysis in Human-Robot Interaction | Notion

Created by Wafa

Project Description

CHRI on Notion

Notion

🗒️ Project Overview

The ROSAnnotator project aims to develop a standalone web application specifically designed to enhance the analysis of Robot Operating System Bag (ROSBag) data, with a particular emphasis on Human-Robot Interaction (HRI). ROSBags are a crucial element in robotics research, serving as a standard format for logging and replaying messages within the ROS ecosystem. These logs can include a wide variety of data types, such as video streams, 3D point clouds, and custom messages tailored for HRI studies. The application is envisioned to be a versatile tool for researchers, enabling the loading of multiple ROSBags, integration with auto-transcription tools for audio data processing, and the provision of a synchronized, interactive dashboard for intuitive data visualization.

🥷 Background

Human-Robot Interaction (HRI) is a growing field of study that explores how humans interact with robots in various contexts, from industrial applications to personal assistance and beyond. The analysis of HRI data is complex, involving multiple modalities such as visual data, audio communications, and sensor data from the robot. The ROS ecosystem provides a flexible framework for robot development and research, but the analysis of ROSBag data, especially from HRI experiments, requires specialized tools. Existing tools like Elan offer some capabilities for annotation and analysis but may not fully meet the unique needs of HRI research, such as handling specific ROS data types or synchronizing multiple data streams.

🎯 Objective

The goal of this project is to implement ROSAnnotator; a standalone web application designed to facilitate the analysis of ROSBag data, with a special focus on Human-Robot Interaction (HRI) captured in various formats, including video streams, 3D point clouds, and custom HRI messages. The application will support loading multiple ROSBags, connecting to an auto-transcription tool for processing audio data, and providing a synchronised, interactive dashboard for data visualisation. Users will be able to manually annotate interactions with custom scales, annotating states (e.g., "child looking at the robot") and events (e.g., "child entering input on the table"). The project will also explore the potential for automatic annotation leveraging audio transcripts.

✏️ Analysis

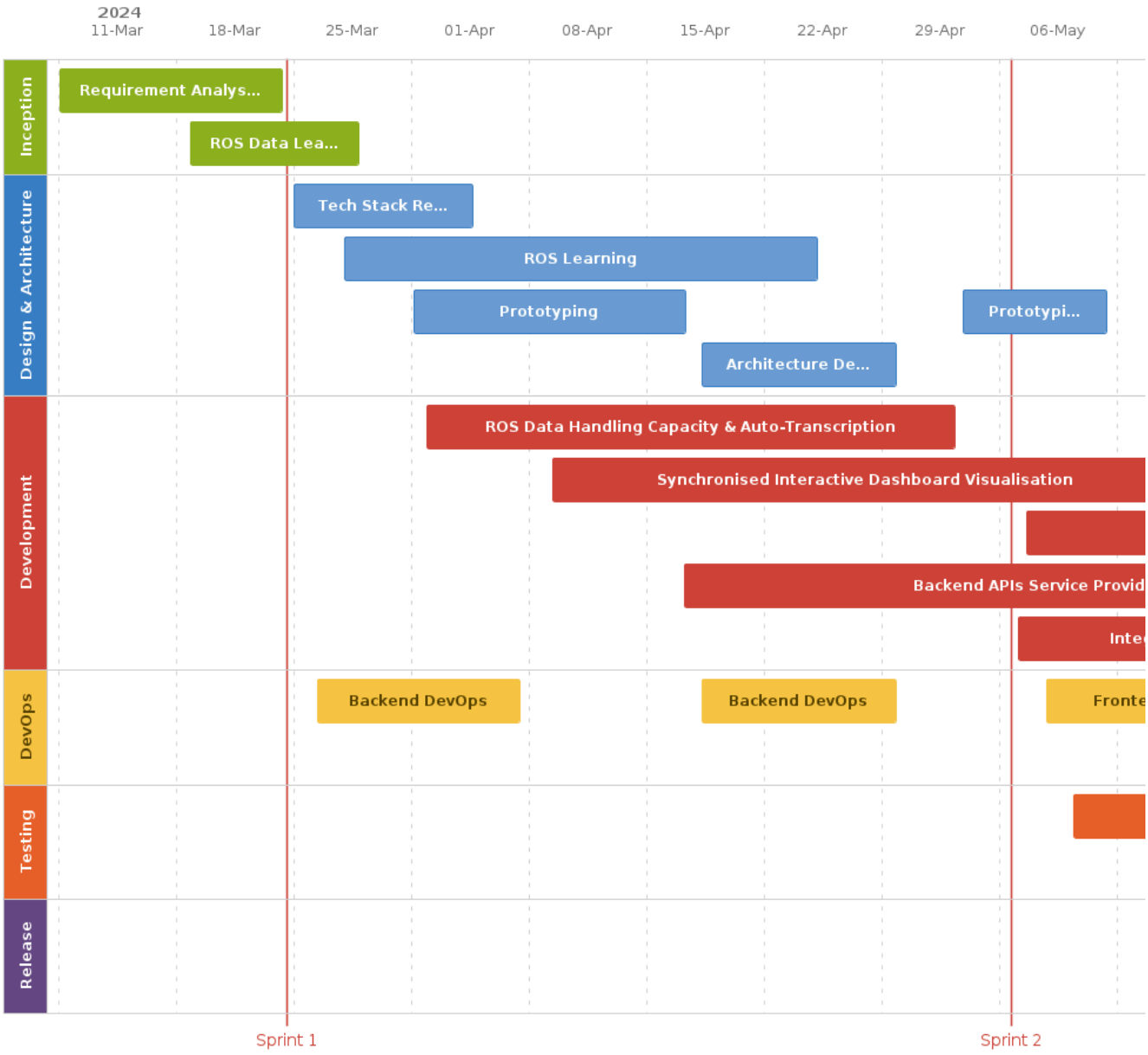
Motivational model	Personas	User Stories	Use Cases	Untitled

📊 Success metrics

Goal	Metric

Data Handling Capabilities	Built robust support for reading and parsing the diverse data types contained in ROSBags, ensuring researchers can access and analyse all relevant data components of their HRI experiments.
Synchronised Interactive Dashboard	Developed an intuitive, user-friendly interface that displays various data streams in a synchronised manner, allowing researchers to interact with and analyze data more effectively.
Annotation Features	Enable detailed annotations of HRI interactions, including custom scales for states (e.g., "child looking at the robot") and events (e.g., "child entering input on the table"), thereby enhancing the depth and specificity of analysis.
Integration and Automation	Integrating with auto-transcription tools for audio data and exploring automatic annotation methods based on audio transcripts are also envisioned to streamline the annotation process and make the tool more versatile.
Non Functional Requirments	Have the app running on any platform and system without heavy manual configuration. The client needs to be able to start the app with only a few simple commands.

🌟 Milestones



⚠ Out of Scope

Due to the complexity of the synchronized data visualization component in the frontend, the handling of complex data types, and the limited timeframe, we have been actively discussing with our client in sprints 1 & 2 to determine what is feasible within our resource constraints. Consequently, we have identified the following five major functional areas to be excluded from the current project scope. Priorities have been assigned to some tasks to ensure appropriate progress.

	Out of Scope Feature	Description	Reference
1	**3D Point Cloud**	All features related to 3D point clouds, including visualization and data processing, will not be implemented.	see user story 2.1

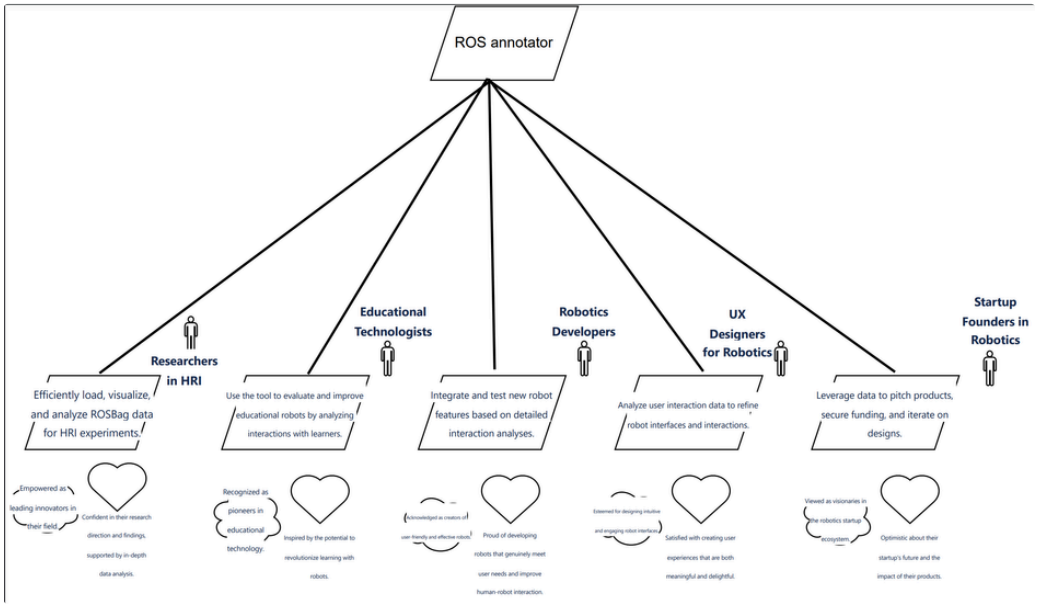
2	**Complex Operations on Frontend Data Display**	The ability to perform complex operations on displayed data in the frontend, such as filtering and toggling data, and video frame manipulation.	see user stories 1.4, and 2.4 - 2.9
3	**Persistent Storage for Processing Data or Annotations**	Interactions with databases for storing the progress of users or processing data will not be included in this project phase.	see user stories 3.6 - 3.7
4	**Batch & Auto Annotations**	The development of features for batch processing and automatic annotations	see user stories 6.1, 6.2, and 6.4
5	**Manual Correction of Auto-Generated Transcripts**	The functionality for manual corrections of automatically generated transcripts	see user story 4.3

These exclusions are aimed at focusing our efforts on deliverables that are achievable within the given constraints, ensuring that we deliver quality and value where it matters most.

Motivational model

Goal Model

Reference: DO-BE-FEEL



DO-BE-FEEL

DO-BE-FEEL table for ROSAnnotator Project

Reference： Motivational model

Users	DO (Functional Goal)	BE (Quality Goal)	FEEL (Emotional Goal)
Researchers in HRI	Efficiently load, visualize, and analyze ROSBag data for HRI experiments.	Empowered as leading innovators in their field.	Confident in their research direction and findings, supported by in-depth data analysis.
Educational Technologists	Use the tool to evaluate and improve educational robots by analyzing interactions with learners.	Recognized as pioneers in educational technology.	Inspired by the potential to revolutionize learning with robots.
Robotics Developers	Integrate and test new robot features based on detailed interaction analyses.	Acknowledged as creators of user-friendly and effective robots.	Proud of developing robots that genuinely meet user needs and improve human-robot interaction.
UX Designers for Robotics	Analyze user interaction data to refine robot interfaces and interactions.	Esteemed for designing intuitive and engaging robot interfaces.	Satisfied with creating user experiences that are both meaningful and delightful.
Startup Founders in Robotics	Leverage data to pitch products, secure funding, and iterate on designs.	Viewed as visionaries in the robotics startup ecosystem.	Optimistic about their startup's future and the impact of their products.

Personas

Note:

- We've developed four personas for this project, tailored to its specialized use scenario, as the initiative is primarily aimed at research purposes.
- You can explore detailed information about each persona by clicking on their names.
- Clicking on the "User End Goal" will reveal their expected outcomes and what they hope to achieve with this product.
- The final column provides references to a list of user stories that may be applicable to each persona.

Persona	Name	End Goal	User Story
	**Dr. Emily Nguyen**	User Persona 1 | :light_ bulb_on: Expectation	1.1 ~ 1.2 2.1 ~ 2.2 3.1 ~ 3.2 5.1
	**Alex Rivere**	User Persona 2 | :light_ bulb_on: Expectation	1.1 ; 1.3 ; 1.4 2.2 ~ 2.4 3.2 ~ 3.4 5.2
	**Samira Campbell**	User Persona 3 | :light_ bulb_on: Expectation	1.1 ~ 1.2 2.5 ~ 2.7 3.5 ~ 3.6 4.1 ~ 4.3

	**Rahul Sharma**	User Persona 4 | :light_bulb_on: Expectation	1.3 2.1 ; 2.2 ; 2.8 ~ 2.9 3.3 ; 3.7 ~ 3.8 5.2 ~ 5.3 6.1 ~ 6.4

User Persona 1

Persona name	**Dr. Emily Nguyen**
Persona role	Assistant Professor
Job description	Dr. Nguyen is an Assistant Professor in Robotics Engineering, specializing in Human-Robot Interaction (HRI) research, focusing on educational robots and their impact on children's learning.

🏛️ Institution

Institution name	TechnoFuture University
Institution size	Medium (500-1000 employees)
Industry	Education & Research in Robotics
Institution Description	TechnoFuture University, nestled in a bustling urban area, stands at the forefront of innovation in education and research, particularly in the field of robotics engineering. It offers a fertile ground for pioneering work in Human-Robot Interaction (HRI).

👥 Demographic information

Age	35
Gender	Female
Income	$80,000 annually
Education level	Ph.D. in Robotics Engineering
Residential environment	Urban

🖋️ Personal quote

- "Understanding the subtle cues in human-robot interaction can revolutionize educational technology."

📜 Biography

Dr. Emily Nguyen has been fascinated by robotics since her undergraduate years. After completing her Ph.D., she dedicated herself to research in HRI, specifically focusing on how robots can enhance learning experiences for children. She balances her time between teaching, conducting research, and contributing to academic journals. Emily is always on the lookout for innovative tools to better analyze and interpret interaction data between children and robots.

Professional goals	Motivators
• To pioneer research that improves robot design for educational purposes. • To develop methodologies for assessing and enhancing robot-child interaction.	• Passion for robotics and education. • Desire to make a significant contribution to the field of HRI. • The quest for finding efficient tools to streamline research processes.
Challenges	**Sources of information**
• Limited time due to teaching responsibilities. • Need for a comprehensive tool that simplifies the analysis of complex HRI data.	• Academic journals and conferences in robotics and HRI. • Online forums and communities dedicated to robotics research. • Collaboration with other researchers and industry professionals.

💡 Expectation

1. Professional ROS Data Analysis:

- Ability to efficiently load and parse ROSBag data containing diverse data types from HRI experiments, particularly those involving educational robots and children.
- Streamline process for identifying and analyzing human-robot interaction that can impact children's learning.

2. Synchronized Display of Multimodal Data:

- Seamless integration and synchronized display of video streams, 3D point clouds, and custom HRI messages to facilitate comprehensive analysis of robot-child interactions.
- High-performance visualization tools that can handle complex datasets without significant lag or processing delays.

3. Intuitive and Customizable Annotation:

- User-friendly interface for manual annotation of interactions, enabling detailed documentation of states (e.g., "child engaged with the robot") and events (e.g., "learning milestone achieved").
- Flexibility in defining and modifying annotation scales to suit specific research needs and objectives.

4. Data Preservation and Export:

- After completing annotations, the expectation is to easily save the annotated datasets in ROSBag format for future usage, sharing with the research community and school, or further analysis.
- Assurance of data integrity and the preservation of all original and annotated data elements within the ROSBag format.

5. Enhanced Research Productivity:

- Reduction in the time and effort required to analyze complex HRI data, allowing for more focus on deriving meaningful insights and developing educational robots.
- A tool that fits seamlessly into the research workflow, complementing existing methodologies and enhancing the ability to conduct rigorous scientific studies.

User Persona 2

Persona name	**Alex Rivere**
Persona role	Ph.D candidate
Job description	Alex is a Ph.D. candidate researching the impact of social robots in healthcare settings, with a focus on elderly care. His work involves understanding how interactions with robots can enhance engagement and companionship for the elderly.

🏛️ Company

Company name	HealthTech Robotics Lab
Company size	Small (50-100 employees)
Industry	Healthcare & Robotics Research
Company Description	The HealthTech Robotics Lab is a pioneering research facility specializing in the intersection of healthcare and robotics. It is dedicated to developing advanced robotic systems designed to improve the lives of the elderly. The lab is renowned for its development of day-care robots that offer companionship and support to seniors living alone, addressing critical challenges in elderly care with cutting-edge technology.

👥 Demographic information

Age	29
Gender	Male
Income	$30,000 annually (stipend)
Education level	Pursuing Ph.D. in Robotics and Healthcare Interaction

Residential environment	Urban, shared apartment near the university

🖋️ Personal quote

- "Robots have the potential to transform elderly care, making companionship accessible to all."

📜 Biography

Alex's journey into robotics began during his undergraduate studies in biomedical engineering. Inspired by the possibilities of improving lives through technology, he decided to pursue a Ph.D. focusing on the role of robots in elderly care. His research aims to identify factors that promote positive engagement between the elderly and robots, hoping to improve adherence to therapy and overall quality of life. Alex is dedicated to his research but faces challenges in analyzing complex interaction data efficiently.

Professional goals	**Motivators**
• To identify key factors that promote positive engagement between the elderly and social robots. • To contribute to the development of robots that can provide companionship and support to the elderly.	• A desire to improve the quality of life for the elderly. • The challenge of integrating technology and healthcare in meaningful ways. • The opportunity to contribute to groundbreaking research in HRI.
Challenges	**Sources of information**
• Analyzing complex and voluminous interaction data within limited time frames. • Finding user-friendly tools to facilitate the comparison of data across multiple sessions and participants.	• Academic journals in robotics, healthcare technology, and gerontology. • Conferences and workshops on robotics and healthcare innovations. • Online communities and forums dedicated to robotics research and healthcare technology.

💡 Expectation

1. Professional ROS Data Management:

- Alex expects the ability to efficiently load and analyze multiple ROSBag datasets from various scenarios collected in his lab.
- This feature is essential for conducting comparative analyses across different sessions and identifying patterns or discrepancies in the interaction data.
- The tool should accommodate the complexity and volume of data specific to elderly care scenarios, enabling streamlined analysis workflows.

2. Advanced Visualization Tools:

- He seeks advanced visualization tools capable of displaying data in a manner that allows for easy identification of key moments and patterns.
- Alex needs the capability to view 3D models of the environment to better understand the spatial dynamics between the elderly and robots, especially in contexts where the robot's mission includes preventing collisions with obstacles.
- The ability to select specific time points for detailed examination is crucial for dissecting interactions and environmental navigation.

3. Annotation and Improvement Tracking:

- Alex requires a user-friendly interface to annotate moments when the robot's performance did not meet expectations.
- This functionality should enable him to document instances of suboptimal interaction or navigation, facilitating a focused review and iteration process.

- The goal is to leverage these annotations to refine robot behaviour, ensuring safer and more effective companionship and support for the elderly.
- The tool must allow for easy modification and tracking of these annotations over time to measure progress and implement enhancements effectively.

User Persona 3

Persona name	**Samira Campbell**
Persona role	UX Designer
Job description	UX Designer at a large robotics company, specializing in designing intuitive interfaces and interactions for industrial robots to enhance safety and productivity on the factory floor.

🏢 Company

Company name	RoboTech Innovations
Company size	Large (Over 5,000 employees)
Industry	Robotics & Manufacturing
Company Description	RoboTech Innovations is a leading company in the field of industrial robotics, known for its commitment to manufacturing the most usable and safe robots on the market. Situated in a technologically advanced industrial park, RoboTech Innovations is dedicated to pushing the boundaries of robotics engineering, focusing on creating robots that enhance productivity and user experience in industrial settings.

👥 Demographic information

Age	32
Gender	Female
Income	$95,000 annually
Education level	Master’s Degree in Human-Computer Interaction

Residential environment	Suburban, owns a home

🎤 Personal quote

"Designing for humans means understanding every gesture and glance. Our robots need to be partners, not obstacles."

📜 Biography

Samira Campbell has always been passionate about the intersection of technology and human interaction. With a background in computer science and a master's in human-computer interaction, she embarked on a career in UX design focused on robotics. At RoboTech Innovations, Samira is tasked with making industrial robots not just tools, but collaborative partners for workers. Her challenge is to design systems that are both efficient and intuitive, ensuring safety and productivity in high-stakes environments. She relies on detailed HRI data to understand user needs and improve design continuously.

Professional goals	**Motivators**
• To innovate in the field of UX design for robotics, making machines more accessible and intuitive for human workers. • To leverage data from HRI studies to inform design decisions and enhance robot-user interaction.	• A strong belief in the potential of robots to transform industries and improve human work environments. • The desire to make technology accessible and useful for people of all skill levels. • The challenge of translating complex robotic capabilities into simple, intuitive interactions.
Challenges	**Sources of information**
• Accessing detailed, actionable HRI data to inform design decisions. • Balancing the technical capabilities of robots with the intuitive expectations of human users. • Ensuring safety and productivity through design in diverse industrial environments.	• Industry conferences on robotics and human-computer interaction. • Journals and publications on UX design, robotics, and ergonomics. • User feedback sessions and field studies in industrial settings.

💡 Expectation

1. Detailed Interaction Visualisation:

- Ability to review video recorded from her company's robots frame by frame, enabling her to observe users' reactions and report on the UX quality of new robots.
- This granular insight is crucial for identifying subtle nuances in human-robot interaction that can significantly impact the user experience.

2. Expert Annotation Capabilities:

- Facility to annotate different moments of the video and audio using expert domain knowledge on UX.
- This includes labelling interactions to feed into future machine learning training, enhancing the robot's behaviour and future user experience.
- The ability to easily categorize and label data for analysis and training purposes is vital for refining robot responsiveness and intuitiveness.

3. Audio Transcript Conversion:

- Access to audio transcripts of interactions, allowing for a detailed analysis of the robot's verbal responses.
- This feature is essential for identifying which replies may be inappropriate or confusing, enabling targeted improvements to robot communication strategies to enhance overall user experience.

User Persona 4

Persona name	**Rahul Sharma**
Persona role	Co-founder and HCI Engineer at AutoTransTech
Job description	Co-founder and HCI Engineer at an innovative startup, AutoTransTech, which develops advanced, completely human-free parcel stations powered by robotics for sorting, transporting, and preparing goods.

🏛️ Company

Company name	AutoTransTech Startup
Company size	Small (30 employees)
Industry	Logistics and Robotics in the Automotive Transportation Sector
Company Description	AutoTransTech is an innovative startup transforming the logistics industry through the development of fully autonomous parcel stations. Leveraging advanced robotics and cutting-edge human-computer interaction principles, the company specializes in creating efficient, human-free systems for sorting, transporting, and preparing goods. AutoTransTech is at the forefront of redefining automotive transportation logistics to meet the evolving demands of global commerce and e-commerce growth.

👥 Demographic information

Age	40
Gender	Male
Income	Variable, dependent on startup success and funding rounds
Education level	Bachelor's Degree in Mechanical Engineering, self-taught in programming and business management
Residential environment	Urban, in a tech-centric city area

🖋️ Personal quote

"Revolutionizing logistics with robotics, we're creating seamless, efficient, and entirely autonomous parcel stations."

📜 Biography

Rahul Sharma transitioned from a mechanical engineer to an entrepreneur with a vision for transforming the logistics industry. At AutoTransTech, Rahul leads the development of autonomous parcel stations that utilize sophisticated robotics to handle sorting and transportation tasks without human intervention. This venture represents a bold step into the future of automotive transportation, where efficiency, speed, and reliability are significant. Rahul's deep involvement in HCI ensures that the interactions between the robots are as seamless and efficient as possible, focusing on optimizing the system's overall performance and reliability.

Professional goals	**Motivators**
• To innovate in the logistics industry by introducing fully autonomous parcel stations that redefine efficiency and reliability. • To overcome the challenges of designing highly efficient, cooperative robotic systems for logistics applications. • To leverage technology for creating scalable, future-proof solutions in logistics, meeting the growing demands of e-commerce and global trade.	• A deep desire to make a positive impact in the field of robot automation industry through technology. • The entrepreneurial challenge of building a successful business that delivers value to society. • The technical challenge of developing robots that are both reliable and commercialisable for business.
Challenges	**Sources of information**
• Balancing rapid development and iteration with thorough and insightful analysis of HRI data to inform design decisions. • Limited resources for dedicated data analysis personnel and tools. • Ensuring the robots are reliable for independent workflow and continue add value to business.	• Industry reports and research papers on robotics and automation within logistics. • Feedback and performance data from the robotic systems in operation. • Conferences, workshops, and seminars focused on robotics technology, and human-computer interaction.

💡 Expectation

1. Behavioural Analysis Capability:

- Rahul expects the ability to use the ROSAnnotator to analyze and study the behaviour of the robots in the parcel stations, identifying areas for optimization and enhancement.

2. Synchronized Multi-Robot Data Visualization:

- He requires the functionality to load multiple ROS data streams from different robots into one synchronized dashboard, enabling the visualization of their cooperative efficiency and performance.

3. Custom Annotation Tools:

- The system should allow for the creation of custom labels for annotating ROS data.
- Rahul aims to define his own scales and event types specific to AutoTransTech's operations, facilitating a tailored analysis approach.

4. Batch and Automatic Annotation:

- Given the high volume of data generated by the robots, Rahul needs the capability for batch and automatic annotation of multiple robots' data simultaneously. This feature is crucial for efficient data processing and analysis.

5. Future AI Integration:

- Looking ahead, Rahul plans to integrate AI into AutoTransTech's systems. The ROSAnnotator's ability to support this integration is essential, as it will enable more sophisticated analysis and predictive modelling of robot behaviour and station efficiency.

User Stories

User Story Table

Please refer to the user story table for a detailed breakdown of our requirements into specific user stories.

Epics

We categorize our user stories and development functional areas using epics. We have divided the project requirements into 7 epics, labelled with IDs from 1 to 7. Each epic represents a cohesive functional area with its own responsibilities. For each epic, both frontend and backend cooperation is required. For a complete view of all epics, please refer to the User Story Table and the "Epic Board" list on our Trello.

Story Points

Justification

We assign story points to each user story primarily based on the difficulty level of implementation. We use 3 and 5 as the base units for forming points. Larger stories are estimated by adding the repetitions of these numbers together, following this sequence: 3, 5, 6, 8, 11, 13, 15, 16, etc. (*e.g. 16 = 3 x 2 + 5* x *2*). We do not assign more than 16 points to a single story; if necessary, we break it down into smaller stories.

Each story is further divided into smaller, actionable tasks assigned to individual team members during a sprint. The total story points for a user story are simply the sum of all the story points of its subtasks.

The reason we choose to use this approach is because this method helps manage workload, adjust velocity, and track the completion of points by each developer more easily, as tasks in this project can vary significantly in difficulty.

Please visit our Product Backlog list on Trello to view the user story distributions and the Sprint Task Tracking Board to see the task breakdown.

Story Point Assignment Approach:

For ongoing tasks within a sprint, if we find that the initial estimation significantly deviates from the actual workload, we adjust the story points accordingly. For future sprints, we first estimate the number of subtasks for each story and then use the Planning Poker method to estimate the points for each task according to their difficulties during a planning meeting. The estimation that receives the highest vote will be assigned.

Velocity Estimation

Background

The project team is engaged in an Agile development process, with sprints as the primary time management structure for delivering new features and updates. The team's current velocity—an estimate of the amount of work they can complete in a single sprint—is a critical factor for planning and forecasting future work.

Justification

Using the smallest story points as a unit, we estimate that a task worth **3 points** will take a developer one week to complete. With approximately 5 developers actively working on the project each week, this amounts to **3 x 5 = 15** points per team per week. Each development sprint lasts 4 weeks, so we estimate that about **60 points** can be completed per sprint.

Please refer to the "Sprint Management Doc -> Sprint Retro" for velocity adjustments.

⚠️ Priority

All priorities are assigned and discussed with our client during meetings. From top to low, the priorities range from the most expected / necessary features that the client wants to the features that can be omitted. We aim to address all `MUST HAVE` features, complete some `COULD HAVE` features, and, if time permits or the effort is minimal, we will tackle some `NICE TO HAVE` features.

Please see the table below for the priority legend.

Priority Legend	
`MUST HAVE`	Essential features that are critical for the software's basic functionality and core objectives.
`COULD HAVE`	Desirable features that enhance user experience or functionality but are not critical for the initial release.
`NICE TO HAVE`	Features identified as lowest priority, excluded from the current development cycle but potentially revisited later.

User Story Table

Note:

- For the end users, please refer back to the list of Personas.
- We do not have points assigned for “Out of Scope“ features (indicates as `OUT`), and may not have points assigned for every lowest priority feature.
- The blue rows indicate the name of each Epic.

Epic ID	User Story ID	As a <User>	I want to <Do Something>	So that <achieve goals>	Velocity Estimation (Story Points)	Priority
ROS & HRI Data Handling Capability						
1	**1.1**	Dr. Emily Nguyen Alex Rivere Rahul Sharma	efficiently **load and parse** various ROSBag data containing diverse data types, such as video streams, and audios *(Note: 3D point clouds is out of scope feature)*	I can efficiently access and analyze the interaction data from the HRI experiments without manual data conversion	15	MUST ...
	1.2	Dr. Emily Nguyen	handle data variabilities in ROSBags, such as **different** message **types** and **formats**	work with data from various sources without needing to preprocess extensively	8	MUST ...
	1.3	Alex Rivere Rahul Sharma	handle **large volumes** of ROSBag data efficiently, ensuring quick loading and parsing times	quickly find relevant data points for my analysis	6	NICE T...
	1.4	Alex Rivere	**filter and search** within the loaded ROSBags for specific data types or messages	I can quick allocte / lookup a recipe	OUT	COULD...
	1.5	Dr. Emily Nguyen Rahul Sharma	have robust support for parsing and visualizing custom **HRI messages** `hri_msgs`	I can deeply understand robot-child interactions and their impacts on learning	OUT	NICE T...
Synchronised Interactive Dashboard						
2	**2.1**	Dr. Emily Nguyen Rahul Sharma	**visualize 3D point clouds** from ROSBag data within the application	analyze spatial aspects of human-robot interactions in educational settings	OUT	NICE T...
		Dr. Emily Nguyen	load and play **video streams** from ROSBags	observe and analyze the dynamics of robot-child interactions in real-time	3	MUST ...

	2.2	Dr. Emily Nguyen Alex Rivere Rahul Sharma	all visualized data streams (video) to be **synchronized** in **real-time** on the dashboard *(Note: 3D point clouds, custom HRI messages are out of scope features)*	observe and analyze how different elements of human-robot interactions interplay simultaneously	8	MUST ...
	2.3	Alex Rivere	**interact** with the dashboard, such as **pausing**, **rewinding**, and **fast-forwarding** through data streams	closely examine specific moments or interactions of interest	15	MUST ...
	2.4	Alex Rivere	**toggle** the **visibility** of different data types on the dashboard	focus on specific aspects of the data without distraction from other information	OUT	COULD...
	2.5	Samira Campbell	review video recordings from industrial robots **frame by frame** within the application	closely observe and analyze users' reactions and interactions with the robots	OUT	COULD...
	2.6	Samira Campbell	**adjust playback speed** and navigate through video recordings easily	focus on critical moments of interaction without missing any details that could inform design improvements	OUT	COULD...
	2.7	Samira Campbell	**zoom** and enhance features on video footage	examine intricate details of user expressions and movements that might indicate usability issues or areas for enhancement	OUT	NICE T...
	2.8	Rahul Sharma	visualize **spatial relationships** and distribution among robots in a shared space	optimize layout and assignments for increased productivity	OUT	NICE T...
	2.9	Rahul Sharma	**highlight** and focus on data **for specific data types** or from specific robots or tasks	drill down into detailed analyses of individual performances within the context of the team's overall operation	OUT	NICE T...
Annotation Features						
3	**3.1**	Dr. Emily Nguyen	**intuitive** and **interactive** interface for annotating video streams and 3D point clouds	precisely mark and note relevant interactions between children and robots without extensive technical training	11	MUST ...
	3.2	Dr. Emily Nguyen Alex Rivere	easily **save** and **export** annotated **datasets** in a common format (e.g., ROSBag)	share findings with the academic community, preserve data for future analysis, or integrate insights into my research workflow	6	MUST ...
	3.3	Alex Rivere Rahul Sharma Samira Campbell	manually annotate the data(video and audio) on a **timeline** with **detailed labels** using my domain knowledge, such as "user	categorize interactions for in-depth analysis and future machine learning model training	5	MUST ...

			confusion", "successful interaction", or "navigation issue"			
	3.4	Alex Rivere	easily **edit** or **delete** annotations	refine my analysis and keep my data organized and accurate as my understanding evolves	5	MUST ...
	3.5	Samira Campbell	annotate the **transcribed text** with **custom** tags or notes	highlight and categorize important verbal interactions or phrases relevant to my research	6	COULD...
	3.6	Samira Campbell	**track** changes and **updates** to **previous annotations** over time	monitor the evolution of user interaction trends and the impact of future modifications	OUT	NICE T...
	3.7	Rahul Sharma	flexibility to **modify** and **expand** my **annotation schema** as our understanding of optimal robot behavior evolves	ensuring our analytics capabilities grow with our company	OUT	NICE T...
	3.8	Rahul Sharma	**store custom annotation** sets within the system	promoting a standardized approach to analyzing and discussing robot performance across our development	6	NICE T...
	3.9	Dr. Emily Nguyen	**synchronize annotations** across different data streams (video, audio transcripts, 3D point clouds)	maintain a cohesive understanding of interactions and events	8	MUST ...
Integration with Auto-Transcription						
4	**4.1**	Samira Campbell	automatically **transcribe audio data** from the ROSBags into text transcripts	analyze verbal interactions without the need for manual transcription, saving time and increasing accuracy	6	COULD...
	4.2	Samira Campbell	**correlate** transcribed text with **specific moments** and events **in the video**	I can gain insights into the verbal aspects of human-robot interactions	6	COULD...
	4.3	Samira Campbell	option to **manually review** and **correct** the auto-transcribed text	ensuring high accuracy and reliability of the data for analysis	OUT	NICE T...
Multiple ROSBags Loading						
5	**5.1**	Dr. Emily Nguyen	manage and **switch** between **multiple ROSBags** easily within the application	I can conduct comparative and composite analysis without significant downtime	8	COULD...
	5.2	Alex Rivere Rahul Sharma	easily **load multiple ROSBags** into the application	perform comparative analyses across different datasets to identify patterns or discrepancies in HRI data.	OUT	COULD...
	5.3	Rahul Sharma	**compare** different **data** format from different operational scenarios **side by side**	assess how well our robots work together under varying conditions	OUT	NICE T...

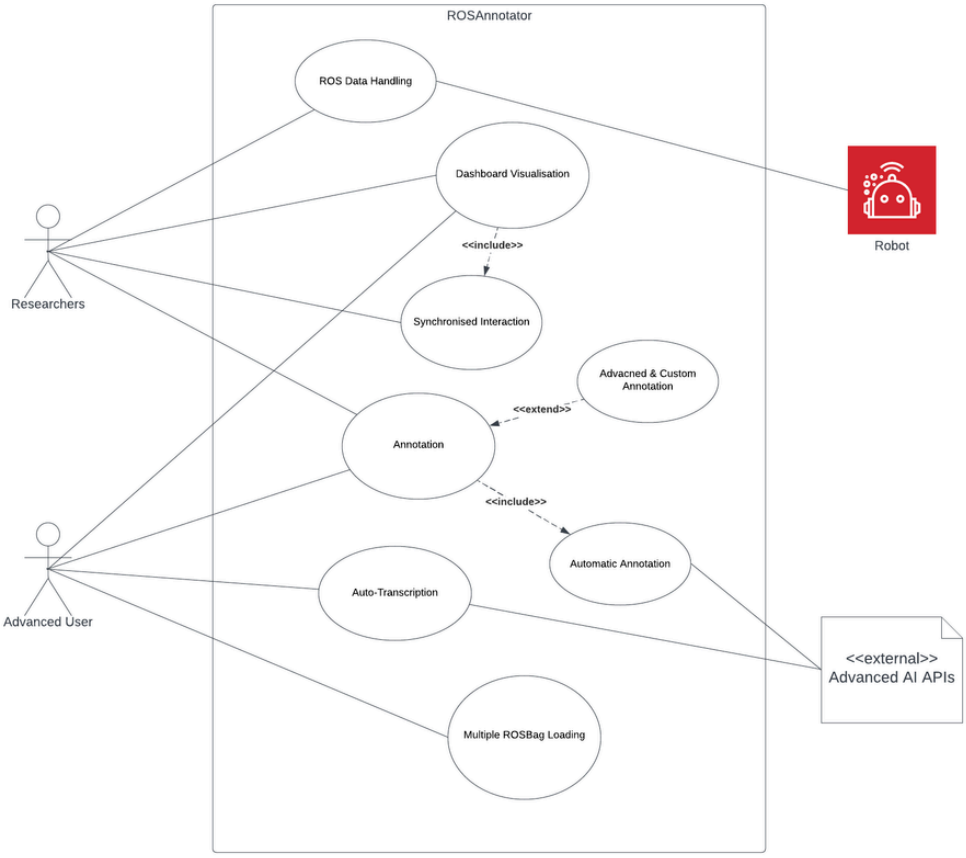
				and identify areas for synchronization improvements		
Automatic Annotation						
6	**6.1**	Rahul Sharma	**batch annotation** capabilities to efficiently process large volumes of data	allowing for quick categorization and analysis of robot performance across multiple parcel stations	OUT	NICE T...
	6.2	Rahul Sharma	**automatic annotation** of common events and behaviors detected in ROS data	streamline the initial stages of data analysis and focus on in-depth evaluation of anomalies and optimization opportunities	OUT	NICE T...
	6.3	Rahul Sharma	**customize rules** for automatic annotation	learn and adapt to our specific operational metrics and goals	6	NICE T...
	6.4	Rahul Sharma	**review and adjust** automatically generated annotations	ensuring accuracy and relevance to our constantly evolving understanding of optimal robot behavior	OUT	NICE T...
Non-Functional Requirements						
7	**7.1**	All	the app can be run on **any platform** without heavy manual configuration	I can use any device to run the system	15	MUST ...
	7.2	All	start using the app with only **a few simple commands** after pull down from GitHub	I can easily start using the app without complex configuration	11	MUST ...

Use Cases

Main Functional Requirements

Use Case	Step & Flow	Priority
ROS Data Handling	ROS Data Handling	MUST HAVE
Dashboard Visualisation	Dashboard Visualisation	MUST HAVE
Synchronised Interaction	Synchronised Interaction	MUST HAVE
Annotation Features	Annotation Features | 1. Basic Annotation	MUST HAVE
Custom Annotation	Annotation Features | 2. Advanced Customisation and Labeling	COULD HAVE
Automatic Annotation	Annotation Features | 3. Automatic Annotation	NICE TO HAVE
Auto-Transcription	Auto-Transcription	COULD HAVE
Multiple ROSBags Loading	Multiple ROSBags Loading	COULD HAVE

UseCase Diagram



Note:

- The two primary actors only represent different groups of users, rather than corresponding to specific roles.

ROS Data Handling

Title: Efficient ROSBag Data Loading and Parsing

Actors: Dr. Emily Nguyen, Alex Rivere, Rahul Sharma

Preconditions: ROSBag data is available and accessible by the application.

Main Flow:

1. The user selects the ROSBag file they wish to load into the application.
2. The application parses the ROSBag data, recognizing and notifying the user of various data types (e.g., video streams, 3D point clouds, audio).
3. The application efficiently loads the data onto the dashboard without any errors, making it readily accessible for analysis.
4. The user accesses and analyzes the HRI data, focusing on interactions without needing manual data conversion.

Postconditions: The user has successfully loaded and parsed ROSBag data, ready for in-depth analysis.

Dashboard Visualisation

Title: Visualizing 3D Point Clouds and Video Streams

Actors: Dr. Emily Nguyen, Rahul Sharma

Preconditions: 3D point cloud and video stream data are loaded into the system

Main Flow:

1. The user selects the ROSBag file containing 3D point cloud and video stream data ready for display.
2. The application loads and displays the 3D point cloud and video streams on the dashboard in a way that the user can interact with the data.
3. The user analyzes spatial aspects and dynamics of human-robot interactions by observing the synchronized data.
4. The user can click on each section of the window to change focus and drag the video or 3D objects to change the viewpoints.

Postconditions:

- All data has been displayed aligned to the same time points, the dashboard interface is intuitive and easy to navigate.
- The user has visualized and analyzed the 3D spatial and video data in synchronization.

Synchronised Interaction

Title: Interacting with the Synchronized Dashboard

Actors: Alex Rivere

Preconditions: Multiple data streams of different types are visualized on the dashboard.

Main Flow:

1. The user views synchronized data streams (video, 3D point clouds) on the dashboard.
2. The user interacts with the dashboard, using controls to pause, rewind, or fast-forward through the video data. Meanwhile, all other data streams (including 3D objects and audio) change correspond to the synchronised timeline.
3. The user focuses on specific moments or interactions of interest for closer examination.

Postconditions: The user has interactively examined specific data points across synchronized data streams.

Annotation Features

1. Basic Annotation

Title: Annotating Video Streams and 3D Point Clouds for State and Event

Actors: Dr. Emily Nguyen

Preconditions: Video and 3D point cloud data are loaded onto the dashboard and in a state ready to be modified.

Main Flow:

1. The user selects a segment of the video or a section of the 3D point cloud for **event** annotation. The application highlights the selected parts and prompts for actions.
2. The user chooses the annotation options, optionally adds in text, and inserts an annotation for the highlighted selection, noting relevant interactions or features.
3. The application shows the message of annotated success or failure.
4. Annotations are synchronized across different data streams for cohesive analysis.
5. The user clicks “save” to save or download the annotated data for future reference or sharing.
6. The system pops up the loading window and prompts for successful download.

Alternative Flow:

1. * The user selects a particular frame of the video or one timepoint of the 3D point cloud for **status** annotation. The application highlights the selected parts and prompts for actions.

Postconditions:

- The user has annotated and synchronized notes across multiple data types.
- The newly labelled data has been downloaded in the ROSBag format for future use.

2. Advanced Customisation and Labelling

Title: Customizing Annotated Datasets

Actors: Alex Rivere, Rahul Sharma

Preconditions: Video and 3D point cloud data are loaded onto the dashboard and in a state ready to be modified.

Main Flow:

1. Refer to Basic Annotation Steps 1 & 2.
2. The user selects the option to customise an annotation with self-defined scales or specific state names or events.
3. (Optional) The application prompts the user to save the self-defined label into the system.
4. The user chooses a labelled timepoint and easily edits or deletes annotations to refine their analysis.
5. The annotated dataset is saved in a common format, like ROSBag, for future loading or analysis.
6. The user shares the annotated dataset with the academic community or team members.

Postconditions: Customized annotations have been added and saved for further analysis or integration into research workflows.

3. Automatic Annotation

Title: Annotating Video Streams and 3D Point Clouds for State and Event

Actors: Dr. Emily Nguyen

Preconditions: Video and 3D point cloud data are loaded onto the dashboard and in a state ready to be modified.

Main Flow:

1. The user selects a segment of the video or a section of the 3D point cloud. The application highlights the selected parts and prompts for actions.
2. The user selects the “Automatic Annotation“ option on the toolbar in the application.
3. The system pops up the loading window to inform the user the application is conducting AI analysis on the selected portion.
4. The application displays and highlights all the newly labelled time points and indicates the automatic annotation has finished.
5. The user can select one or more annotations to refine the labelling. Actions include editing, deleting and adding new annotations.
6. Refer to Basic Annotation Steps 4 - 6.

Postconditions:

- The user has annotated and synchronized notes across multiple data types.
- The newly labelled data has been downloaded in the ROSBag format for future use.

Auto-Transcription

Title: Transcribing Audio Data from ROSBags

Actors: Samira Campbell

Preconditions: ROSBag with audio data is loaded into the application.

Main Flow:

1. The user selects the ROSBag file containing audio data for transcription.
2. The application displays the data as an audio waveform in the dashboard.
3. The user selects the “Auto-transcription“ button on the toolbar.
4. The application pops up the loading window and prompts the user that “AI is Calculating”.
5. The system automatically transcribes the audio data into text transcripts and displays the transcript in an independent window for further actions.
6. The user can review and, if necessary, correct the auto-transcribed text for accuracy. The user clicks the “Confirm” button to insert the audio transcript.
7. The transcribed text is correlated with video and 3D point cloud data in the synchronised timelines for comprehensive analysis, enhancing the user's analysis capabilities.

Postconditions:

- Verbal interactions from the ROSBag are transcribed and displayed on the dashboard.
- The data is aligned with other data types on the dashboard for future annotations.

Multiple ROSBags Loading

Title: Managing and Analyzing Multiple ROSBags

Actors: Dr. Emily Nguyen, Alex Rivere, Rahul Sharma

Preconditions: Multiple ROSBags are available for analysis.

Main Flow:

1. The user selects multiple ROSBags for loading into the application.
2. The application efficiently loads the selected ROSBags, ready for analysis.
3. The user selects one displaying option from various view options, including side-by-side, synchronised, controlled variables, and so on.
4. The user performs comparative analysis across the different datasets, identifying patterns or discrepancies.
5. The user performs annotations on multiple datasets. Refer to Auto-Transcription
6. Insights from the comparative analysis inform further research or operational improvements.

Postconditions: The user has successfully managed and analyzed multiple ROSBags, gaining valuable insights for research or operational enhancements.

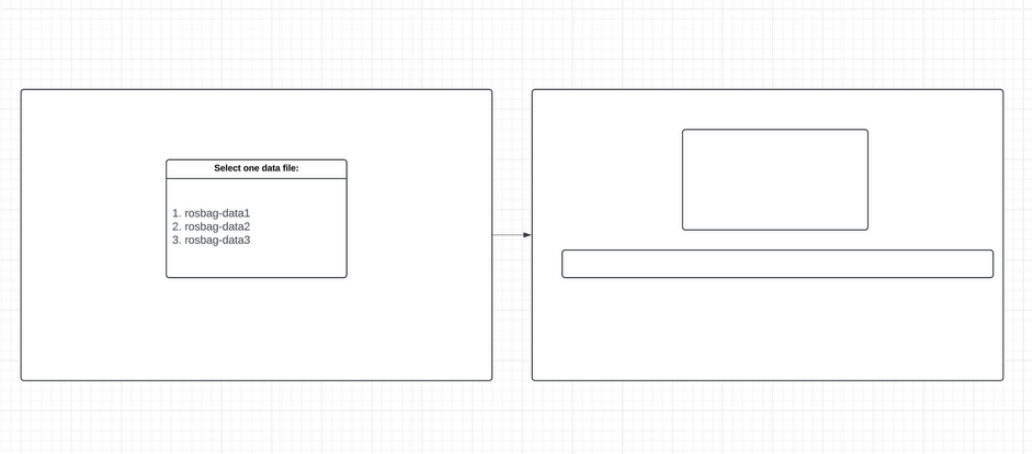
Prototype

https://www.figma.com/file/xUryyBhgdlA6OjlhQglWT5/Untitled?type=design&node-id=20%3A206&mode=design&t=4VONMFhcNv1gw6oI-1

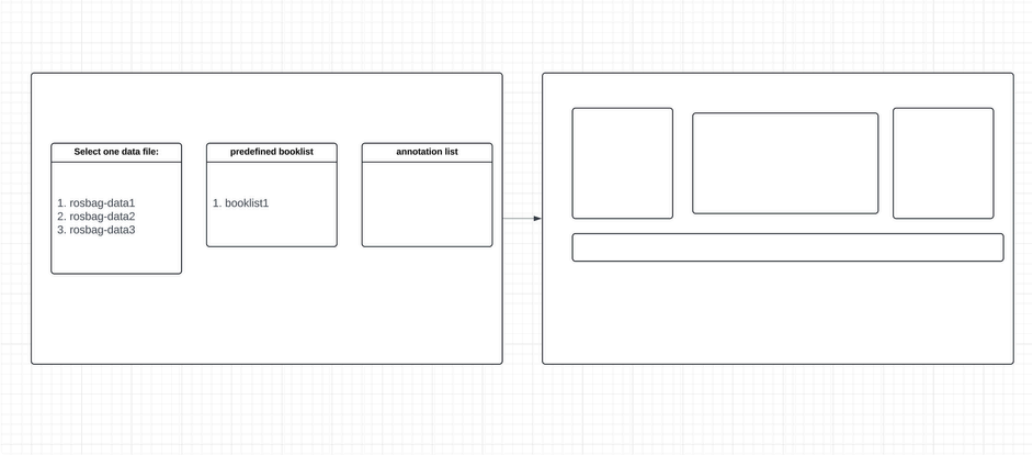
Additional Helper Page

🧠 Idea Perceptions:

1.

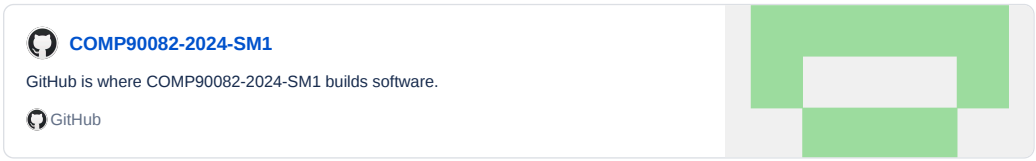


2.



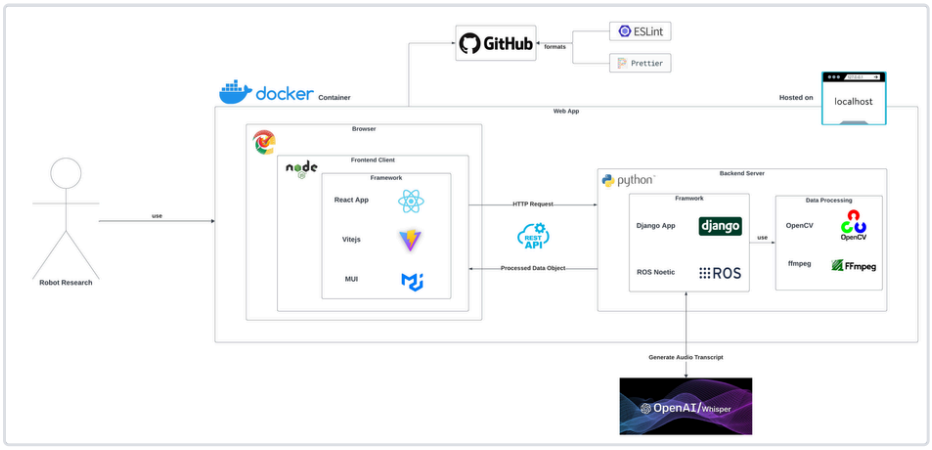
💻 Development

GitHub Link:



Architecture Diagram

📐 Architecture



Architecture flow

Please check our demonstration video:
https://drive.google.com/file/d/19A8k8uDG5NgB5xpRQiOwexR_nYtFT-9I/view?usp=sharing Connect your Google account

🚀 Deployment strategy

Deployment region: localhost

Instructions: Refer to our Deployment Instructions doc.

⌨️ Rest API details

Refer to our API Documentation doc.

Workflow Guide

Introduction

This guide aims to streamline our team's use of GitHub for managing software development projects. It covers essential Git practices, naming conventions, and workflow strategies to ensure consistency, efficiency, and clarity throughout our projects.

1. Git Workflow

- **Cloning a Repository**: Start by cloning the repository to your local machine using `git clone <repository-url>`.
- **Branching Strategy**:
  - Always create a new branch for your work, no matter how small.
  - Use the naming conventions outlined above.
  - Ensure you are branching off from the correct base branch (usually `main`).
- **Commit Messages**:
  - Write clear, concise commit messages in the imperative mood, e.g., "Add payment processing module."
  - Begin the message with a capital letter and do not end it with a period.
  - Use the body of the commit message to explain "what" and "why", not "how".

2. Pull Requests (PRs) and Code Reviews

- **Creating Pull Requests**:
  - PRs should be made against the `main` branch.
  - Include a clear title and a detailed description of the changes.
  - Reference related issues using hashtags, e.g., `Fixes #123`.
- **Code Review Process**:
  - At least one team member must review the PR before it can be merged.
  - Reviewers should check for code quality, adherence to project standards, and overall integration.
  - Use GitHub's commenting features to ask questions or suggest improvements.

3. Merging and Deployments

- **Merging PRs**:
  - Use the "Squash and Merge" option for small features or fixes to keep the history clean.
  - Use "Merge Commit" for significant features where preserving the detailed history of changes is valuable.
- **Deployment**:
  - After merging, changes should be tested in a staging environment before deploying to production.
  - Use tags and releases in GitHub to manage deployment versions.

4. Handling Issues

- **Creating Issues**:
  - When encountering a bug or a needed improvement, create an issue.
  - Provide a descriptive title and a detailed description including steps to reproduce, expected outcome, and actual outcome.
- **Assigning Issues**:
  - Assign issues to the appropriate team member.
  - Tag issues with labels such as `bug`, `feature`, `urgent`, etc., to help categorize and prioritize.

5. Additional Resources

- Git Documentation
- GitHub Flow

Frontend Documentation

The frontend of our web-app is built upon Node.js and Vite, which enable rapid development with its extensive built-in features, fast development process, and provides a solid foundation for our web-app development.

Use this page to navigate through different part of frontend documentations

💻 Frontend Tech Stack A comprehensive guide for setting up the frontend environment and tech stack used.	Development Environment
🗒️ Coding Standards Frontend development obeys certain coding standards, please read through this document before contributing to the project	Frontend Coding Standards

Development Environment

This document outlines the development environment setup for our project, which utilizes Node.js and React for frontend development, and Python with Django for the backend. This guide is intended to help new contributors get their development environment up and running smoothly.

Frontend Development

Node.js and React

Our frontend application is built using React, a popular JavaScript library for building user interfaces, particularly single-page applications. Node.js is used for running JavaScript code server-side, and npm (Node Package Manager) is used to manage project dependencies.

Prerequisites:

- **Node.js**: Ensure you have the latest LTS (Long-Term Support) version of Node.js installed, which includes npm.
- **Vite**:
  - **Purpose**: Vite is a modern, fast frontend build tool that significantly improves the development experience by providing instant server start and hot module replacement.
  - Start the development server provided by Vite:

    `npm run dev` accessible at `localhost:3000` (your localhost).

Core Libraries:

1. **React**:
   - **Description**: React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It lets you compose complex UIs from small and isolated pieces of code called “components”.
2. **React Player**:
   - **Description**: A React component that provides a simple player for various types of media including video and audio. It supports multiple streaming sources and provides a uniform API across different browser implementations.
3. **Material-UI (@mui/material and @mui/system)**:
   - **Description**: Material-UI provides a robust, customizable, and accessible library of foundational and advanced components, enabling you to build your own design system and develop React applications faster.
4. **Styled-components**:
   - **Description**: Allows you to write actual CSS code to style your components. It removes the mapping between components and styles, enabling complete and complex component styles within your JavaScript files.
5. **React LRC**:
   - **Description**: React LRC is a React component designed to display lyrics synchronized with audio or video playback. It is ideal for projects that require dynamic lyric or caption synchronization with media content, enhancing the multimedia experience.

Frontend Directory Structure:

- `src/`: Contains all source files.
- `components/`: Contains reusable React components.
  - `Block/`: Manages annotation blocks on the timeline.
  - `CustomSnackbar/`: Provides feedback through snackbar notifications.
  - `Timeline/`: Handles the video timeline and interactions.
  - `Transcript/`: An autoscroll transcript that synchronizes with the video.

- `App.tsx`: The entry component that integrates various components.
- `main.tsx`: Responsible for rendering the React application to the DOM.

Styles Management

This project uses styled-components for styling the React components, allowing for scoped and reusable style definitions.

Example of Styled Component:

```
const ScrollableTimelineContainer = styled('div')({
  overflowX: 'scroll',
  overflowY: 'hidden',
  width: '100%',
  cursor: 'pointer',
  padding: '10px 0',
});

```

Key Components

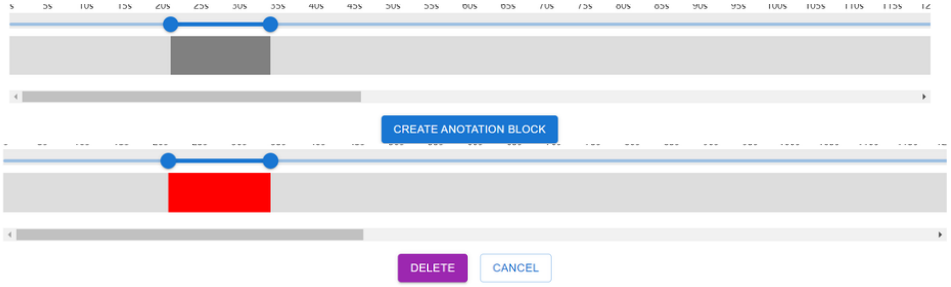
Each component serves a specific purpose in the application, facilitating modularity and maintainability. Here's an overview of some critical components:

Timeline Component

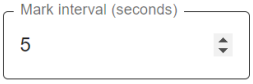
The Timeline component is the central interface for video navigation and annotation management, providing essential functionality for interacting with video content and annotations.

- **Features**:
  - **Navigation and Seeking**: Users can navigate through the video by moving a draggable slider along the timeline.

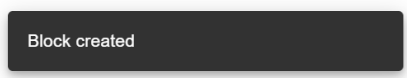
  - **Annotation Interaction Block**: Supports the addition, selection and deletion of annotations block directly on the timeline.



  - **Customizable Mark Interval**: Allows users to adjust the interval at which marks appear on the timeline, enabling a more tailored view according to the specific needs of the video or annotations.



  - **Utilizes Snackbar**: Provides user feedback for actions such as creating, adjusting, or deleting annotations via the CustomSnackbar component.



Blocks Component

The Block component represents annotations on the timeline, providing visual and interactive markers that the user can manipulate to annotate specific parts of the video.

- **Features**:
  - Interactivity: Blocks can be created with a double slider, allowing the user to adjust their start and end times directly on the timeline.

  - **Selection and Deletion**: Selecting a block reveals options for resizing or removing it, with deletions confirmed via the Snackbar.（Displayed in the timeline section）
  - **Color Coding**: Blocks are color-coded to indicate their selection status, enhancing visual clarity and user interaction.（Displayed in the timeline section）

CustomSnackbar Component

The CustomSnackbar provides essential feedback to users, confirming the results of their interactions within the Timeline. It plays a critical role in enhancing user experience by providing immediate, contextual information on actions performed.

- **Features**:
  - **Feedback on User Actions**: It displays notifications for various user actions like adding, deleting, or modifying Blocks. This helps confirm that the intended actions have been successfully completed.
  - **Error Handling**: The Snackbar also serves to inform the user of any errors or issues that might occur during interaction, ensuring clear communication of system states.（Refinement in sprint3）
  - **Customizable Appearance**: The duration and appearance of the Snackbar messages are customizable, allowing alignment with the overall design and user experience strategy of the application.（Refinement in sprint3）

Transcript Component

The Transcript component is an autoscroll screen that contains the transcript of the video. The transcript will scroll automatically in-sync with the video and adjusts to timeline changes accordingly.

- **Features**:
  - **Autoscroll**: The Transcript automatically scrolls according to the video timeline.
  - **Highlighted Current Line**: The line that is currently being spoken is highlighted in blue in accordance with the video player and moves onto the next line when finished.

Integrated Functionality

The design ensures seamless interaction between components:

Timeline and Blocks: The Timeline controls the rendering and manipulation of Blocks, handling their placement based on user actions and video playback.
Timeline and CustomSnackbar: The Timeline triggers the Snackbar for providing feedback related to annotation actions, ensuring users receive immediate and relevant feedback on their interactions.

Frontend Coding Standards

1. **Folder Structure**
   - **Directory Per Component:** Each React component should have its own directory with the same name as the component.
     - Example: `UserProfile/UserProfile.tsx`
   - **Test Files:** Place test files next to their corresponding component and use `.test.tsx` suffix.
     - Example: `UserProfile/UserProfile.test.tsx`
   - **Styles:** If using CSS modules, the styles file should also be named after the component.
     - Example: `UserProfile/UserProfile.module.css`

2. **File Naming Convention**
   - **Components:** Use PascalCase for React components. Files should match the name of the default export.
     - Example: `UserProfile.tsx`
   - **Hooks:** Prefix custom hooks with `use` and use camelCase.
     - Example: `useUser.ts`
   - **Utilities and Helpers:** Use camelCase for utility files.
     - Example: `arrayHelpers.ts`
   - **Type Definitions:** Use PascalCase and include `Type` suffix if it helps in clarity, otherwise just PascalCase.
     - Example: `UserProfileType.ts`, `Theme.ts`

3. **Function and Method Naming Convention**
   - **Regular Functions:** Use camelCase for function names. The name should start with a verb expressing what the function does.
     - Example: `calculateAge`, `fetchUserData`
   - **Event Handlers:** Start with `handle` followed by the action and the element.
     - Example: `handleButtonClick`, `handleChange`

3. **Functionality**
   - **Clarity and Descriptiveness:** Names should clearly and accurately describe their functionality or role.
   - **Consistency:** Apply naming conventions consistently across the project.
   - **Simplicity:** Prefer simplicity over complexity in naming. Avoid abbreviations unless widely understood.

Backend Documentation

The backend of our web-app is built upon Django, which enable rapid development with its extensive built-in features, scalability to handle increased data, and provides a solid foundation for our web-app development.

Use this page to navigate through different part of backend documentations

💻 Backend Deployment Instruction A comprehensive guide for setting up the development environment and workflow for the backend of the project	Deployment Instructions
🔌 API Documentation A list of provided APIs with parameter requirements and response format.	API Documentation
🛠️ Function Documentation Functions used within each API are listed with its details.	Function Documentation
🗒️ Coding Standards Back-end development obeys certain coding standards, please read through this document before contributing to the project	Backend Coding Standards

Deployment Instructions

This document provides a comprehensive guide for running (or setting up) the development environment and workflow for the backend of our project. It includes instructions for preparing all the `ROS` dependencies, starting the backend `Django` server, developing within the environment, and placing `rosbag` files.

- The backend is running on Python version **3.9** & Django version **4.2**
- The `rosbag` analysis is done by using **ROS1 Noetic** and related Python packages provided by RoboStack. Please refer to the ROS Environment section for more details.
- Base image for Docker: `ubuntu:jammy` (Ubuntu 22.04)

Prerequisites

Docker Installation

- Before you begin, ensure that you have Docker installed on your machine. If not, download and install Docker from Docker's official site.
- Verify that the **Docker daemon** is running by executing `docker info` in your terminal. This should return information about the Docker client and server. If not, please start the Docker daemon.

Backend Directory Structure

- `ros_annotator/`: The main app with the same name as the project.
- `rosbag_processing/`: The app for ROSbag data processing tasks.
- `processed_data/`: This folder is used to store all processed files, including audio, video, waveform, and transcription.
- `rosbag-data/`: This folder is for user to store their ROSbag files.
- `manage.py`: The Django command-line utility.
- `env_config/`: This folder contains environment-specific configuration settings

Starting the Backend Server

To start the backend server, follow these steps:

1. **Navigate to the Backend Directory**
   Change to the backend folder by running:

```
cd src/backend
```

2. **Start the Server**
   Use Docker Compose to start the backend services:

```
docker-compose up --build
```

This command will build the image using `Dockerfile` from ground up and start all the required backend services as defined in the `docker-compose.yml` file. It also installs all the `ROS` dependencies.

> *Note: The image size is about 4.84 GB, so it may take a while for the first time to download and build the image. The speed depends on your computational resources and internet connection.*

Use the below command to check the image and the status of running container:

```
# you should see a image with the name 'backend:ros-annotator'
docker image ls
# you should see a running container with the name 'ros-annotator'
docker ps
```

3. **Access the Backend Server**

Once the server is up and running, you can access the backend server at `http://localhost:8000/`. You should see the Django REST framework interface.

> *Note: The backend server is running in development mode, and the backend code base has been map to the container through Docker volume, so it will automatically reload whenever you make changes to the code.*

Visit `http://localhost:5678/` for debugging purposes.

Development Setup Guide & Work Flow

To develop on the backend, proceed with the following steps (or follow the gif below for a visual guide):

1. **VSCode Extensions**: Ensure you have the Dev Container extension installed in your VSCode. The Dev Container extension allows you to develop inside a Docker container.
2. **Attach to the Running Container**: In VSCode, use the Dev Container extension to attach to the running Docker container. This can typically be done through the command palette (Ctrl+Shift+P or Cmd+Shift+P) and selecting "Attach to Running Container."
3. **Install Necessary Extensions in Dev Container**: Within the dev container environment in VSCode, install the Python (and Jupyter extensions if you are going to run Jupyter Notebook) to facilitate development. (*Note: This only need to be done once when you attach the container for the first time.*)

Using ROSbag Datas

To work with `rosbag` files:

- **Data Placement**: Place your rosbag data into the `src/backend/rosbag-data` folder. This location is set up to be accessible within the backend environment using docker volume.

API Documentation

Functionality APIs

1.Data Processing APIs (/rosbag)

Path	/process_rosbag
Type	POST

Request	<pre>{
	  "bag_filename": "example.bag",
	  "booklist_filename": "booklist.json",
	  "annotation_filename": "annotations.csv"
	}</pre>

Response	On Success:
	<pre>{
	  "video_path": "<relative_path_to_video>",
	  "audio_path": "<relative_path_to_audio>",
	  "waveform_image_path": "<relative_path_to_waveform_image>",
	  "audio_transcript": "<transcript_text>",
	  "booklist_data": { "Gesture ": ["wave gesture", "encourage gesture"]
	  "annotation_data": [{ "id": 1, "axisType": "type", "annotationBlocks
	  "message": "Processing complete"
	}</pre>
	On Error:
	<pre>{
	  "error": "No bag_filename provided"
	}</pre>
	<pre>{
	  "error": "Invalid request method"
	}</pre>
	<pre>{
	  "error": "Invalid bag filename"
	}</pre>
	<pre>{
	  "error": "Invalid booklist filename"
	}</pre>
	<pre>{
	  "error": "Invalid annotation filename"
	}</pre>
	<pre>{
	  "error": "Error loading annotation data"
	}</pre>

Path	/list_filenames
Type	GET
Request	None

Response	On Success:
	`{ "rosbag_files": ["file1.bag", "file2.bag"], "booklist_files": ["booklist1.json", "booklist2.json"], "annotation_files": ["annotation1.csv", "annotation2.csv"] }`
	On Error:
	`{ "error": "Invalid folder paths" }`

Path	/update_booklist
Type	POST
Request	`{ "name": "booklist.json", "data": { "key 1": [list 1], "key 2": [list 2] } }`

Response	On Success:
	`{ "message": "Booklist updated successfully", "New BookList": { "key 1": [list 1], "key 2": [list 2] } }`
	On Error:
	`{ "error": "Booklist file does not exist" }`

Path	/save_annotation
Type	POST
Request	`{ "annotation_name": "annotation", "annotation_data": [ { "id": 1, "axisType": "type",`

```
      "axisName": "name",
      "axisBooklistName": "booklistName",
      "annotationBlocks": [
        {
          "start": 0,
          "end": 10,
          "text": "example"
        }
      ]
    }
  ]
}
```

Response

On Success:

```
{
  "message": "Annotation data saved successfully"
}
```

On Error:

```
{
  "error": "Invalid JSON format"
}
```

Path	/get_annotation
Type	GET

Request

```
{
  "annotation_filename": "annotations.csv"
}
```

Response

On Success:

```
{
  "annotation_data": [
    {
      "id": 1,
      "axisType": "type",
      "axisName": "name",
      "axisBooklistName": "booklistName",
      "annotationBlocks": [
        {
          "start": 0,
          "end": 10,
          "text": "example"
        }
      ]
    }
  ]
}
```

On Error:

```
{
  "error": "File not found"
}
```

Function Documentation

This document serves as a reference for the functions used within each APIs

API: /process_rosbag

1. **extract_images_from_rosbag**
   - **Purpose**: Extract images from a ROS bag file containing RGB image messages and save them to the specified output folder.
   - **Inputs**: `bag_filename` (str), `output_folder` (str)
   - **Outputs**: None
   - **Dependencies**: `cv2`, `Bag` from `rosbag.bag`, `CvBridge`
2. **extract_video**
   - **Purpose**: Convert extracted images into a video file.
   - **Inputs**: `bag_filename` (str), `output_folder` (str)
   - **Outputs**: None
   - **Dependencies**: `subprocess`, `ffmpeg`
3. **extract_audio**
   - **Purpose**: Extract audio data from the ROS bag file and save it as an MP3 file.
   - **Inputs**: `bag_filename` (str), `output_folder` (str)
   - **Outputs**: None
   - **Dependencies**: `Bag` from `rosbag.bag`
4. **plot_waveform**
   - **Purpose**: Plot the waveform of an audio file and save it as an image.
   - **Inputs**: `audio_path` (str), `output_filename` (str), `start_sec` (float, optional), `end_sec` (float, optional)
   - **Outputs**: None
   - **Dependencies**: `AudioSegment` from `pydub`, `matplotlib`
5. **combine_video_audio**
   - **Purpose**: Combine video and audio files into a single video file.
   - **Inputs**: `output_folder` (str)
   - **Outputs**: None
   - **Dependencies**: `subprocess`, `ffmpeg`
6. **generate_srt**
   - **Purpose**: Generate an SRT file containing speech transcripts with timestamps.
   - **Inputs**: `uri` (str), `output_folder` (str)
   - **Outputs**: `srt_file_path` (str)
   - **Dependencies**: `speech_v1p1beta1` from `google.cloud`, `format_time`
7. **format_time**
   - **Purpose**: Format a duration into HH:MM:SS,MMM format.
   - **Inputs**: `duration` (datetime.timedelta)
   - **Outputs**: Formatted time string (str)
   - **Dependencies**: None
8. **encode_file_base64**
   - **Purpose**: Encode a file as a base64 string.

- **Inputs**: `file_path` (str)
- **Outputs**: Encoded base64 string (str)
- **Dependencies**: `base64`

9. **get_relative_path**
   - **Purpose**: Extract the relative path of a given file path based on a predefined base path.
   - **Inputs**: `path` (str)
   - **Outputs**: `relative_path` (str)
   - **Dependencies**: `os`
10. **load_annotation**
    - **Purpose**: Load and parse annotation data from a CSV file located in a specific directory, and return the structured annotations.
    - **Inputs**: `annotation_filename` (str)
    - **Outputs**: `annotations` (list)
    - **Dependencies**: `os`, `csv`

Backend Coding Standards

1. **Folder Structure**
   - All src files should reside in a `src/` directory
   - Create and manage the Django app based on functionality needed
   - Use descriptive folder names to represent the main functionality of the module.

2. **Naming Convention**
   - **Files**: snake_case
   - **Functions**: snake_case
   - **API endpoints**: snake_case

3. **Functionality**
   - Each function should only be responsible for a single task
   - Use descriptive function names that clearly convey their purposes
   - Keep solution simple

4. **Error Handling**
   - Return HTTP status codes consistent with the nature of the error (e.g., `400` for bad requests, `405` for invalid request methods)
   - Implement appropriate error handling mechanisms for all potential failure points, including file I/O operations and data processing

Technologies Used

1. **CVBridge**

A ROS (Robot Operating System) package that provides a bridge between ROS image messages and OpenCV image formats. It allows ROS nodes to efficiently convert image messages from ROS topics into OpenCV-compatible data structures, and vice versa. This enables seamless integration of ROS-based robotic systems with OpenCV-based computer vision algorithms and applications. CVBridge is commonly used for tasks such as image processing, object detection, navigation, and perception in robotic applications.

2. **FFmpeg**

A powerful multimedia framework that can decode, encode, transcode, mux, demux, stream, filter, and play almost any type of audio and video files. It is widely used for tasks such as video and audio editing, format conversion, streaming media, and batch processing.

3. **OpenCV**

A library of programming functions mainly aimed at real-time computer vision. It provides tools and algorithms for tasks such as image and video processing, object detection and tracking, facial recognition, and machine learning-based vision applications. OpenCV is widely used in various fields including robotics, augmented reality, medical imaging, and surveillance.

4. **Whisper**

A library for converting spoken language into text. It utilize machine learning algorithms to analyze audio data and transcribe it into written text. This is useful for application that need automatic transcription of audio recordings.

Testing Documentation

1. Data Loading Test Cases

Test Case 1.1: Load a Single ROSBag

- **Description:** Verify the application can load a single ROSBag file.
- **Steps:**
  a. Open the application.
  b. Select a ROSBag file.
  c. Click "Load."
- **Expected Outcome:** The ROSBag data is loaded and displayed on the dashboard.

Test Case 1.2: Load Multiple ROSBags

- **Description:** Verify the application can load and manage multiple ROSBag files simultaneously.
- **Steps:**
  a. Open the application.
  b. Select multiple ROSBag files.
  c. Click "Load."
- **Expected Outcome:** All selected ROSBag files are loaded and displayed separately on the dashboard.

Test Case 1.3: Handle Corrupted ROSBag

- **Description:** Verify the application handles corrupted or invalid ROSBag files gracefully.
- **Steps:**
  a. Open the application.
  b. Select a corrupted ROSBag file.
  c. Click "Load."
- **Expected Outcome:** An appropriate error message is displayed, and the application remains stable.

2. Auto-Transcription Test Cases

Test Case 2.1: Transcription of Audio Stream

- **Description:** Verify the audio transcription feature works correctly.
- **Steps:**
  a. Load a ROSBag containing an audio stream.
  b. Click "Transcribe."
- **Expected Outcome:** The transcription text appears and aligns with the audio stream.

Test Case 2.2: Handle Missing Audio Stream

- **Description:** Verify the application handles missing audio streams gracefully.
- **Steps:**
  a. Load a ROSBag without an audio stream.
  b. Click "Transcribe."
- **Expected Outcome:** An appropriate error message is displayed.

3. Annotation Test Cases

Test Case 3.1: Manual Annotation

- **Description:** Verify manual annotation capabilities.
- **Steps:**
  a. Load a ROSBag with video/audio data.
  b. Select a time segment.
  c. Add a manual annotation.
- **Expected Outcome:** The manual annotation appears on the dashboard at the specified time.

Test Case 3.2: Automatic Annotation

- **Description:** Verify automatic annotation based on audio transcription.
- **Steps:**
  a. Load a ROSBag with an audio stream.
  b. Click "Transcribe" to generate the transcription.
  c. Click "Auto Annotate."
- **Expected Outcome:** The automatic annotations appear based on the audio transcription.

Test Case 3.3: Edit Annotations

- **Description:** Verify the ability to edit existing annotations.
- **Steps:**
  a. Add or auto-generate an annotation.
  b. Edit the annotation's content or time.
- **Expected Outcome:** The annotation is updated as expected.

4. Dashboard and Synchronization Test Cases

Test Case 4.1: Dashboard Visualization

- **Description:** Verify the dashboard displays synchronized visualizations.
- **Steps:**
  a. Load a ROSBag with audio, video, and other streams.
  b. Play the streams from the dashboard.
- **Expected Outcome:** All data streams are synchronized and visible on the dashboard.

Test Case 4.2: Synchronization of Annotations

- **Description:** Verify annotations synchronize with data streams.
- **Steps:**
  a. Add annotations to a loaded ROSBag.
  b. Play the data streams.
- **Expected Outcome:** The annotations appear at the correct times on the dashboard.

5. Security and Compliance Test Cases

Test Case 5.1: Role-Based Access Control

- **Description:** Verify only authorized users can access sensitive data.
- **Steps:**
  a. Log in as an unauthorized user.
  b. Attempt to access sensitive data.
- **Expected Outcome:** Access is denied, and an error message is displayed.

Test Case 5.2: Data Encryption

- **Description:** Verify data is encrypted in transit.

- **Steps:**
  a. Load a ROSBag file.
  b. Monitor the data being transmitted over the network.
- **Expected Outcome:** Data is encrypted during transmission.

Test Case 5.3: Consent Management

- **Description:** Verify consent management is respected.
- **Steps:**
  a. Load a ROSBag file with missing consent.
  b. Attempt to process the data.
- **Expected Outcome:** An error message is displayed, preventing the use of data without consent.

Testing document Sprint3

Test Case Scenario 1: Basic Operations and Error Handling for Overlapping Blocks

1. **Launch and Load Files**
   - Start the application and load a ROSBAG data file.
   - Click on "Load Me" without selecting a predefined booklist or an annotation file.
2. **Adding Axes and Creating Blocks**
   - Add an axis using the "Add New Axis" button.
   - Attempt to create two blocks on the same axis with overlapping times to test error handling.
3. **Mark Interval Adjustment**
   - Adjust the mark interval slider to a lower setting and verify that the visual representation of blocks adjusts accordingly on the timeline.
4. **Validation of Error Message**
   - Ensure an error message is displayed when attempting to create overlapping blocks.
5. **Close Application**
   - Close the application to ensure no errors are thrown upon exit.

Acceptance Criteria

- The application should prevent the creation of overlapping blocks and display an appropriate error message.
- Adjusting the mark interval should visually alter the size of the blocks on the timeline without errors.

Outcome: PASS

Test Case Scenario 2: Managing Axis Names and Using Booklists

Objective: Test the ability to rename axes, utilize a predefined booklist, and verify data saving and retrieval functionality.

Test Steps:

1. **Launch and Load Files:**
   - Start the application.
   - Select a ROSBAG data file and a predefined booklist.
   - Click on "Load Me".
2. **Axis and Annotation Management:**
   - Add two new axes.
   - Rename the first axis to "Gesture Analysis".
   - On the second axis, set type to "Selected" and select a category from the loaded booklist.
   - Create annotation blocks on both axes.
3. **Data Saving:**
   - Click "Save and Send Data" to export the annotation data.
   - Close and reopen the application, then load the exported annotation file to verify data persistence.
4. **Validation of Data Integrity:**
   - Ensure the loaded annotations match the previously saved states.

Acceptance Criteria:

- Axes should be correctly renamed and retain their new names throughout the session.
- Annotations linked with booklist categories should be accurately displayed.
- Saved data should be correctly restored upon reloading, with all annotations and axis configurations intact.

Outcome: PASS

Test Case Scenario 3: Shortcut Keys and Simultaneous Block Creation

Objective: Evaluate the functionality of shortcut keys for simultaneous block creation on multiple axes.

Test Steps:

1. **Launch and Setup:**
   - Open the application and load necessary ROSBAG and booklist files.
   - Add three axes and assign the same shortcut key ('A') to each.
2. **Using Shortcut Keys:**
   - Press 'A' to create blocks on all three axes at the current timeline marker.
3. **Verification of Block Creation:**
   - Verify that a block has been created on each axis at the exact same time interval.
4. **Error Testing:**
   - Attempt to create overlapping blocks using the shortcut key to check error handling.

Acceptance Criteria:

- The same shortcut key should trigger block creation on all assigned axes simultaneously.
- Overlapping block creation should be prevented, and an error message should be displayed.

Outcome: PASS

Test Case Scenario 4: Comprehensive Workflow with Annotations and Axis Deletion

Objective: Test a comprehensive workflow including file loading, axis management, detailed annotation creation, and axis deletion.

Test Steps:

1. **Initial Setup:**
   - Start the application, load a ROSBAG file, and select an annotation file.
   - Load both files by clicking "Load Me".
2. **Annotation and Axis Handling:**
   - Add several axes and create multiple annotation blocks with varied durations.
   - Use the "Manage Axes" function to delete one axis.
   - Adjust the mark interval and verify the changes reflect visually.
3. **Saving and Revalidation:**
   - Save the session using "Save and Send Data".
   - Reopen the application, load the saved data, and verify all changes except the deleted axis are persisted.

Acceptance Criteria:

- All added axes and annotations should be correctly displayed according to the user's actions.
- Deleting an axis should remove it from the session and not appear upon reloading.
- Changes to the mark interval should dynamically adjust the display of annotation blocks.

Outcome: PASS

Test Case Scenario 5: Simultaneous Annotation Creation via Shortcut Keys

Objective: Verify if the system can successfully create annotation blocks on each axis when multiple axes are set to respond to the same shortcut key.

Test Steps:

1. Launch the application and load a ROSBAG file along with a predefined booklist.
2. Click on "Load Me" to load the data.
3. Add three new axes.
4. Set the same shortcut key (e.g., the 'A' key) for each axis.
5. Press the 'A' key and check if an annotation block is successfully created on each axis.
6. Verify the correct time range for each annotation block and ensure there are no overlaps.
7. Adjust the mark interval slider and observe if the display of blocks on the timeline dynamically adjusts.
8. Click “Save and Send Data” to save the session.
9. Close the application.

Acceptance Criteria:

- Each axis should correctly create an annotation block upon triggering the shortcut key.
- Blocks should not overlap and should be displayed correctly under the adjusted mark interval.
- Data should be saved correctly without any error messages.

Outcome: PASS

Test Case Scenario 6: Dynamic Axis Property Adjustment and Error Handling

Objective: Test the functionality of changing axis types, handling annotation block overlap errors, and deleting axes.

Test Steps:

1. Launch the application, select a ROSBAG file and an annotation file.
2. After loading, add two axes.
3. Attempt to create two overlapping annotation blocks on the first axis to trigger an error message.
4. Change the type of the second axis to “selected” and choose a predefined booklist name.
5. Attempt to delete the first axis.
6. Use a shortcut key to create an annotation block on the second axis.
7. Adjust the mark interval and verify visual changes of blocks.
8. Save and send data, checking for errors.
9. Close the program.

Acceptance Criteria:

- The system should prevent the creation of overlapping blocks and display an appropriate error message.
- Changes in axis type and deletion operations should be executed correctly without system crashes.
- Shortcut keys should trigger annotation block creation correctly.

Outcome: PASS

Test Case Scenario 7: Annotation File Loading and Recovery

Objective: Test whether the system can accurately restore a previous state after loading an annotation file.

Test Steps:

1. Launch the application, load a ROSBAG file, and an annotation file.
2. Verify that axes and annotation blocks are matched with the configuration saved in the annotation file after loading.
3. Add new annotation blocks on each axis using different shortcut keys.
4. Change the text on some annotation blocks.

5. Adjust the mark interval to ensure correct size and positioning adjustments of blocks.
6. Save and send data, verifying the format and content.
7. Close the program.

Acceptance Criteria:

- All axes and annotation blocks should be correctly restored according to the saved configuration in the annotation file.
- Any new modifications should be saved correctly, and the data file's structure and content should be error-free.

Outcome: PASS

Test Case Scenario 8: Loading Only ROSBAG and Annotation File

Objective: Test the behavior of the application when loading only a ROSBAG file and an annotation file without a predefined booklist.

Test Steps:

1. Start the application and load a ROSBAG file along with an annotation file.
2. Verify that axes and annotation blocks are created based on the data from the annotation file.
3. Attempt to create new annotation blocks without selecting a booklist name (since no booklist is loaded).
4. Save and send data to check if the absence of a booklist affects the saving process.
5. Close the application.

Acceptance Criteria:

- The application should successfully load the ROSBAG and annotation files without errors.
- Users should be able to create new annotation blocks, but without the option to select a booklist name.

Test Case Scenario 9: Changing Axis Properties and Saving Data

Objective: Test the functionality of modifying axis properties, creating blocks, and saving data.

Test Steps:

1. Open the application, load a ROSBAG file, and a predefined booklist.
2. Add multiple axes with different axis types and names.
3. Change the shortcut keys for some axes and create annotation blocks using these keys.
4. Modify the axis type and name for a selected axis using the Axis Manager.
5. Adjust the mark interval and verify the visual representation of blocks.
6. Save and send data to check if all modifications and annotations are saved accurately.
7. Close the application.

Acceptance Criteria:

- Changes in axis properties should reflect accurately in the interface and during block creation.
- Modifying the mark interval should update block sizes accordingly.
- Saved data should include all changes and annotations without errors.

Outcome: PASS

Test Case Scenario 10: Block Creation and Axis Type Conversion

Objective: Test block creation and converting an axis type, resulting in clearing block content and changing to a selected block.

Test Steps:

1. Open the application and load a ROSBAG file with a predefined booklist.

2. Add a new axis and create an annotation block on it.
3. Input text into the block using "type in" functionality.
4. Convert the axis type to "selected" using Axis Manager.
5. Verify that the block content is cleared and the axis type changes to "selected."

Acceptance Criteria:

- The block content should be cleared upon converting the axis type to "selected."
- The axis should visually represent a "selected" state after the conversion.

Outcome: PASS

Test Case Scenario 11: Shortcut Key Configuration and Block Creation

Objective: Test configuring shortcut keys and creating blocks using those shortcuts.

Test Steps:

1. Launch the application and load a ROSBAG file along with a predefined booklist.
2. Set shortcut keys for each axis type using Axis Manager.
3. Use the configured shortcut keys to create blocks on corresponding axes.
4. Verify that blocks are created correctly and reflect the axis type.

Acceptance Criteria:

- Shortcut keys should trigger block creation on the respective axes.
- The created blocks should display the correct axis type.

Outcome: PASS

Test Case Scenario 12: Annotation File Loading and Axis Restoration

Objective: Test loading an annotation file to restore axes and annotations.

Test Steps:

1. Start the application and load a ROSBAG file with a predefined booklist.
2. Create multiple axes and annotation blocks on them.
3. Save the annotations as an annotation file.
4. Close the application and reopen it.
5. Load the saved annotation file.

Acceptance Criteria:

- Loading the annotation file should restore all previously created axes and annotations.
- The restored axes should reflect the saved data accurately.

Outcome: PASS

Test Case Scenario 13: Mark Interval Adjustment and Block Visual Representation

Objective: Test adjusting the mark interval and verifying the visual representation of blocks.

Test Steps:

1. Open the application and load a ROSBAG file with a predefined booklist.
2. Create multiple annotation blocks on different axes.
3. Adjust the mark interval slider to a lower setting.

4. Verify that the size of blocks on the timeline adjusts accordingly.

Acceptance Criteria:

- Adjusting the mark interval should visually alter the size of blocks without overlapping or distortion.
- The blocks should maintain their positions relative to the timeline.

Outcome: PASS

Test Case Scenario 14: Error Handling for Overlapping Blocks

Objective: Test error handling when creating overlapping blocks.

Test Steps:

1. Launch the application and load a ROSBAG file with a predefined booklist.
2. Add a new axis and create two blocks with overlapping time intervals.
3. Attempt to save the annotations or perform other actions.

Acceptance Criteria:

- The application should prevent the creation of overlapping blocks and display an appropriate error message.
- Other functionalities should remain unaffected by the overlapping blocks error.

Outcome: PASS

Test Case Scenario 15: Statistical Accuracy of Multiple Axes and Annotations

Objective: Verify the statistical calculations for multiple axes and annotations.

Test Steps:

1. Launch the application and load a ROSBAG file with a predefined booklist.
2. Create multiple axes and add annotation blocks on each axis.
3. Review the statistics for axes and annotations provided by the application.

Acceptance Criteria:

- The statistical data for axes (e.g., total number, types) should accurately reflect the created axes.
- The statistical data for annotations (e.g., total number of blocks, duration) should accurately reflect the created annotations.

Outcome: PASS

Test Case Scenario 16: Edit and Save New Booklist Without Loading

Objective: Test editing and saving a new booklist without initially loading it into the application.

Test Steps:

1. Start the application without loading a predefined booklist.
2. Use the Axis Manager to create new axes and annotations.
3. Edit the booklist within the application.
4. Save the edited booklist and close the application.
5. Verify if a new booklist file is generated with the edits.

Acceptance Criteria:

- The application should allow editing and saving a new booklist even without initially loading it.

- The saved booklist file should contain the edited data and be accessible for future use.

Outcome: PASS

Test Case Scenario 17: Modify Loaded Booklist and Save Changes

Objective: Test modifying a loaded booklist and saving the changes.

Test Steps:

1. Launch the application and load a predefined booklist.
2. Use the Axis Manager to edit the loaded booklist (e.g., change annotation names, add/remove annotations).
3. Save the changes to the loaded booklist.
4. Close the application and reopen it to verify if the modifications are retained.

Acceptance Criteria:

- The application should allow modifications to a loaded booklist.
- Saving the changes should update the loaded booklist file, and the changes should persist after reopening the application.

Outcome: PASS

Test Case Scenario 18: Subtitle and Timeline Synchronization

Objective: Verify the synchronization between subtitles and the timeline with ROSBAG-generated videos.

Test Steps:

1. Load a ROSBAG file with a predefined booklist and annotations.
2. Play the video and observe the subtitles displayed.
3. Verify that the subtitles correspond accurately to the timeline events in the video.

Acceptance Criteria:

- Subtitles should display relevant information corresponding to the events in the ROSBAG-generated video.
- The timeline should match the events in the video, ensuring synchronization between the displayed data and video content.

Outcome: PASS

Test Case Scenario 19: Annotation Table Accuracy with Multiple Axes

Objective: Test the accuracy of the annotation table when multiple axes and annotations are created.

Test Steps:

1. Load a ROSBAG file with a predefined booklist and annotations.
2. Create multiple axes and add annotation blocks on each axis.
3. Review the annotations table to ensure it accurately reflects the created annotations.

Acceptance Criteria:

- The annotations table should display all created annotations across multiple axes.
- Each annotation entry in the table should contain accurate information regarding its axis, duration, and content.

Outcome: PASS

Test Case Scenario 20: Annotation File Creation and Data Persistence

Objective: Test the functionality of creating and saving an annotation file, as well as verifying data persistence upon reloading the application.

Test Steps:

1. Open the application and load a ROSBAG file.
2. Create multiple axes with different types and add annotation blocks to each axis.
3. Save the annotations using the "Save and Send Data" button.
4. Close the application.
5. Reopen the application and load the saved annotation file.
6. Verify that the previously created axes, blocks, and annotations are restored correctly.
7. Make changes to the annotations or add new ones and save the updated data.

Acceptance Criteria:

- The application should allow users to create multiple axes and annotation blocks.
- Saving annotations should generate a valid annotation file (CSV format).
- Upon reloading the application with the saved annotation file, the axes, blocks, and annotations should be restored accurately.
- Users should be able to make changes to the annotations and save the updated data without issues.

Outcome: PASS

Code Review

This guideline assists in documenting automated code reviews effectively on Confluence. It ensures a comprehensive record of what code was reviewed, the criteria for selection, participants, findings, and follow-up actions.

Selection Criteria of Reviewed File

Files with the most centralised logic is selected for review, this includes:

1. Frontend components
2. Frontend pages
3. Backend views
4. Backend models

Code Review Sessions

2024-04-22 Code Review (Peer Review)

2024-05-01 Code Review (Frontend ChatGPT Review)

2024-05-02 Code Review (Backend ChatGPT Review)

2024-04-22 Code Review (Peer Review)

🗓️ Date

Apr 22, 2024

👥 Participants

- @Tianqi Wang
- @yucpeng1
- @Harry Wang

ℹ️ Summary

In this peer-to-peer review, participants evaluated the usability and design of the frontend components of a video editing timeline interface. Key improvements were discussed and decisions made without the involvement of ChatGPT.

🗣️ Topics Reviewed

Topic	Issue	Decision
Frontend “Timeline” Component	The timeline represents the video and needs to be easier for the user to scroll. Currently feels clumsy and too squished.	Use another UI design for the timeline such as a material-ui slidebar. The timeline length on the screen needs to be increased.
Frontend “Block” Component	The “Block” component needs to represent an interval on the timeline. It currently cannot have overlapping start and finish times.	Fix the issue of the block not being able to have overlapping start and finish times. The creation logic of the block needs to be modified to fit research needs.
Frontend “Create block” button	The create block button slides with the sliding timeline and creates an uneasy to use experience for clicking.	The “Create Block” button needs to be placed outside of the sliding window and be at a fixed position regardless of where the timeline is currently at. UI arrangement needs to be fixed on the “Timeline” component.

↪️ Decisions

- Ability to overlap block
- Fix bug so block can have the same start and end time
- Extend timeline length
- Place create block out of sliding timeline

2024-05-01 Code Review (Frontend ChatGPT Review)

📅 Date

May 1, 2024

👥 Participants

- @Bowen Fan
- @yuchsong2

🔗 Pull Request

https://github.com/COMP90082-2024-SM1/ros-annotator/pull/4 Connect your Github account

ℹ️ Summary

In this review of the pull request, ChatGPT GitHub action is used to generate prompts during the review. The main inspected files are the files that contain our Components. We have decided that the most important and logic-intense portion of the code is within each frontend component and the component files ending with ".tsx" are in most need of an automatic review. More specifically, "Timeline.tsx", "Block.tsx" and "Transcript.tsx" are reviewed in this pull request.

ChatGPT has spotted the lack of commenting in some vital areas and praised everything else in the frontend. No improvements are suggested in any other fields including Visual Representation, Structure, Resource, User Input, Logic and so on.

🗣️ Topics Reviewed

Topic	Issue	Decision
Documentation and Comments	There appears to be not enough comments in Transcript.tsx and Block.tsx component according to ChatGPT.	While this is true in some areas of the components such as in some areas in Transcript.tsx, there is not enough supporting arguments to add more comments in Block.tsx as the code are all self-explanatory. Block is a simple rectangular component that represents an interval. We decided to add more commenting within Transcript.tsx and leave Block.tsx as is.
Testing and Edge Case Considerations	ChatGPT mentioned adding testing and edge case considerations to the components to ensure security.	Testing is done seperately from the component files, which is not sent to the ChatGPT prompt with this Github Action. For now, this suggestion is taken in mind and safely set aside.

↪️ Decisions

- Add more comment in Transcript.tsx
- Review testing considerations

💡 Conclusion

While suggesting about commenting and documentation helped us realise some areas of improvements in our codebase, ChatGPT seems to not be able to provide any other suggestions for this frontend review specifically, except for some generic prompts. At the frontend, more manual peer-to-peer reviewed is definitely required to make further improvements in the future, much like the 22/04 review we conducted earlier.

2024-05-02 Code Review (Backend ChatGPT Review)

Date

May 2, 2024

Participants

- @Yujie Zheng
- @Harry Wang

Pull Request

https://github.com/COMP90082-2024-SM1/ros-annotator/pull/5 Connect your Github account

Summary

In this review of the pull request, ChatGPT GitHub action is used to generate prompts during the review. The main inspected files are the files that contain our Views. We have decided that the most important and logic-intense portion of the code is within “views.py” file that contains most of the logic and functions.

Topics Reviewed

Topic	Issue	Decision
File Error Handling	“Ensure that error handling mechanisms are in place for potential exceptions during file processing.”	Add error handling “try” and “except” clauses when handling files, including opening “.ros”, “.mp3”, “.txt” files and so on when processing the intermediate files.
Error Messages	“Consider adding more detailed error messages in the response to provide better feedback to users.”	Following the above topic and issue, add detailed print and error logging messages when encountering an error.
Code Refactoring	“Refactor the code to separate concerns and improve modularity for better maintainability.”	We discussed the possibility of separating the functions into seperate files for ease of reading. However, for now, there is not enough files for this to be practical. File refactoring will be set aside until more content is added.
Documentation and Comments	“Add more detailed documentation to explain the purpose of each function and the overall workflow of the code.”	The documentation on the workflow is not present in the code and is put onto Confluence. However, suggesting that each function to be more detailed is a right direction. Next, we will add docstrings to the most important functions to give more detailed description and comment.

Decisions

- Add error handling when opening files

- Log detailed error messages
- Consider refactoring code into different files in later sprints
- Add docstrings to important functions

2024-05-11 Code Review

🗓️ Date

May 10, 2024

👥 Participants

- @yucpeng1
- @Bowen Fan
- @yuchsong2

🔗 Pull Request

https://github.com/COMP90082-2024-SM1/ros-annotator/pull/8 Connect your Github account

ℹ️ Summary

In this review of the pull request, ChatGPT GitHub action is used to generate prompts during the review. The main inspected files are the files that contain our themes, with the main file 'theme.tsx' inspected. Manual review is largely incorporated into this review as well due to the lack of ChatGPT's ability to visualize how good the theme on the frontend should look like. The main AI-assisted reviewed areas are the code consistency and cluttering.

🗣️ Topics Reviewed

Topic	Issue	Decision
Theme Consistency and Standards	The code snippet uses comments effectively to describe the purpose of each color and style setting. However, the use of inline comments can be inconsistent.	Standardize comment style to improve readability and maintain consistency across the project.
Scalability and Extendibility Messages	The current theme configuration is hard-coded, which might limit flexibility if theme variations are needed in the future.	Consider implementing a theme factory or builder pattern to generate themes dynamically, allowing easier modifications and scalability.
Error Handling	Not applicable directly as this configuration doesn't involve operational logic that might throw errors.	We discussed the possibility of separating the functions into seperate files for ease of reading. However, for now, there is not enough files for this to be practical. File refactoring will be set aside until more content is added.
Documentation and Comments	While the code includes basic comments, there is a lack of comprehensive documentation on how	Add documentation either in the code or in a linked developer's manual on how to modify and extend these themes for different layouts or UI requirements.

	these themes can be extended or overridden.	

Decisions

- Standardize comment style to improve readability and maintain consistency across the project.
- Consider implementing a theme factory or builder pattern to generate themes dynamically, allowing easier modifications and scalability.
- We discussed the possibility of separating the functions into seperate files for ease of reading. However, for now, there is not enough files for this to be practical. File refactoring will be set aside until more content is added.
- Add documentation either in the code or in a linked developer's manual on how to modify and extend these themes for different layouts or UI requirements.

2024-05-16 Code Review

🗓️ Date

May 16, 2024

👥 Participants

- @Tianqi Wang
- @Bowen Fan
- @Yujie Zheng

🔗 Pull Request

https://github.com/COMP90082-2024-SM1/ros-annotator/pull/9 Connect your Github account

ℹ️ Summary

The pull request involves reviewing two React components, `StatisticsTier` and `StatisticsAnnotation`, which display various statistics about data blocks within a given context (`AxesProvider`). These components are integral to providing dynamic statistical data visualizations based on user interactions or data changes.

🗣️ Topics Reviewed

Topic	Issue	Decision
Error Handling	Both components throw an error if not used within an `AxesProvider`. This is critical for ensuring that the components do not function outside their intended context.	The current implementation of error handling is appropriate and should be maintained. Consider adding a custom hook in the future for context checks to reuse across different components.
Performance Optimization	Calculations like average, median, and maximum durations are computed directly in the render method, which could lead to performance issues on large datasets.	This is less relevant as the information given to the calculation is inherently small by size due to the give data structure for our annotation purposes. Trying to fix this issue now will run into problems of premature optimization. This is left unchanged.
Code Duplication	Similar error handling and some logic are duplicated across the two components.	Abstract common functionality into custom hooks or helper functions to reduce duplication and improve maintainability.

⤴️ Decisions

Retain existing error handling while exploring a reusable pattern for context validation.

- Reduce code duplication by introducing custom hooks for shared logic.
- Implement robust data validation to prevent runtime errors due to unexpected data formats.

2024-05-17 Code Review

🗓️ Date

May 17, 2024

👥 Participants

- @Harry Wang
- @Yujie Zheng

🔗 Pull Request

https://github.com/COMP90082-2024-SM1/ros-annotator/pull/10 Connect your Github account

ℹ️ Summary

This review examines a series of functions integrated into a Python module, designed for processing multimedia data from ROS bags and other sources. The functionality spans extracting images and videos, audio transcription, and manipulating file paths and annotations.

🗣️ Topics Reviewed

Topic	Issue	Decision
Error Handling	Several functions like `extract_images_from_rosbag` and `load_annotation` could raise runtime errors that are not explicitly caught within the functions.	Implement robust try-except blocks where appropriate, especially where file operations and external command executions (e.g., `subprocess.run`) are involved.
Performance Optimization	Frequent file operations and subprocess calls can be optimized. For example, reading files repeatedly in a loop or using subprocesses inside frequently called functions might degrade performance.	This is deemed unneccessary as the primary time constraint of concern at stake is the openAI whisper audio to text transcription service. The other workflow are limited by how fast the api can give us back the transcript, and would run into premature optimization.
Code Modularity and Structure	he code spans a wide range of functionalities within a single module which might hinder maintainability as the project scales.	Break down the module into smaller sub-modules, each focusing on specific functionalities like audio handling, video processing, and data management.
Documentation and Code Clarity	Functions like `encode_file_base64`	Enhance documentation within the code, ensuring each function is accompanied by clear, concise comments

	and `get_relative_path` are straightforward but lack inline comments explaining their use and potential edge cases.	explaining its purpose and any assumptions it makes.

Decisions

- Implement additional error handling across all functions to manage exceptions and provide more informative error messages.
- Refactor the code into smaller, more manageable modules specific to their operational contexts.
- Add inline comments for better understanding and maintenance.

2024-05-20 Code Review

🗓️ Date

May 20, 2024

👥 Participants

- @ABHISHEK TUMMALAPALLI
- @Tianqi Wang

🔗 Pull Request

https://github.com/COMP90082-2024-SM1/ros-annotator/pull/11 Connect your Github account

ℹ️ Summary

The pull request reviews the `Timeline` and `AnnotationTable` React components used in a multimedia annotation application. The `Timeline` component provides interactive controls for annotating media based on time, while the `AnnotationTable` displays a summary of annotations. Both components utilize React hooks for state and effect management, styled components for layout, and interact with a shared context for state.

🗣️ Topics Reviewed

Topic	Issue	Decision
Error Handling	The code assumes the presence of `axes` context without null checks in `AnnotationTable` and has minimal error handling for asynchronous operations in `Timeline`.	Implement null checks and more robust error handling, particularly for the network request in the `Timeline` component.
Performance Optimization	The `Timeline` component performs potentially expensive operations like mapping over axes and generating marks directly in the render method, which could lead to performance issues with large datasets.	Use `React.memo` for child components and `useMemo` for derived data to avoid unnecessary recalculations and re-renders. However, at this stage, this optimization is not currently needed.
Code Modularity and Structure	Complex logic within components, such as event handlers and context manipulations,	Refactor complex logic into custom hooks or smaller components to improve readability and reusability.

	could be better organized or abstracted.	
Consistency and Best Practices	Inconsistent use of styling (mix of `styled` components and inline styles) and	1. Standardize styling approach and extract magic numbers and repeated values into constants or theme settings. 2.

Decisions

- Implement additional error handling and context validation to prevent runtime errors and improve user feedback on failure cases.
- Refactor complex component logic to enhance readability and maintainability.
- Standardize and improve styling practices to ensure consistency and ease of maintenance.

2024-05-21 Code Review

🗓️ Date

May 21, 2024

👥 Participants

- @Tianqi Wang
- @Bowen Fan
- @yuchsong2

🔗 Pull Request

https://github.com/COMP90082-2024-SM1/ros-annotator/pull/12 Connect your Github account

ℹ️ Summary

This review assesses the `Transcript` component designed to render a transcript in LRC format, allowing users to interact with individual lines to navigate through a media playback. The component leverages React hooks and styled components for functionality and styling, respectively.

🗣️ Topics Reviewed

Topic	Issue	Decision
Functionality and Interaction	The component allows users to click on transcript lines to update the playback time, but lacks feedback mechanisms such as tooltips or visual cues beyond color changes.	Enhance user interaction by providing visual feedback like hover effects or tooltips to indicate that lines are clickable.
Performance Optimization	The component re-renders potentially expensive operations (like the `lineRenderer` callback) without necessary optimizations.	There is no such constant rendering and optimization here is not necessary.
Accessibility	Current implementation may not fully support accessibility features such as keyboard navigation or screen reader compatibility.	Implement accessibility improvements such as keyboard focusable elements and ARIA attributes to improve usability for all users.
Consistency and Best Practices	The component has commented-out code and	Clean up commented-out code and review the component to ensure all features are implemented and necessary.

	unused hooks that could confuse maintenance efforts or suggest incomplete features.	Ensure hooks and variables are utilized effectively.

Decisions

- Implement user feedback mechanisms such as hover effects or tooltips on transcript lines.
- Optimize rendering of dynamic components using `React.memo` or `useMemo` to prevent unnecessary re-renders.
- Clean up the codebase to remove any unused or commented-out code and ensure all hooks and features are necessary and fully implemented.

Decision Making

Decision	Date
Decision on Speech-To-Text problem Owned by Tianqi Wang • Updated on Apr 10, 2024 Status Complete Impact High Driver @yucpeng1 Approver client Contributors @Bowen Fan @yucpeng1 @Tianqi Wang Informed client Due date Apr 10, 2024 Resources https://comp90082-2024-na-… Confluence Open preview	Apr 10, 2024
Decision on Predefined Booklist Owned by Tianqi Wang • Updated on Apr 24, 2024 Status In progress Impact High Driver @Yujie Zheng Approver Contributors Informed client Due date Apr 23, 2024 Resources \uD83D\uDCD8 Background The client requests a pre-defined booklist for the annotation task. However, as … Confluence Open preview	Apr 23, 2024
Decision on intro page Owned by Tianqi Wang • Updated on Apr 22, 2024 Status Not started / In progress / Complete Impact High / Medium / Low Driver Approver Contributors Informed Due date Resources \uD83D\uDCDA Relevant data \uD83D\uDCD8 Background \uD83C\uDF08 Options considered Option … Confluence Open preview	

Decision log

Create decision

Decision	Status	Stakeholders	Outcome	Due date	Owner
Decision on Predefined Booklist	IN PROGRESS			Apr 23, 2024	
Decision on intro page	NOT STARTED / IN PROGRESS / COMPLETE				
Decision on Speech-To-Text problem	COMPLETE			Apr 10, 2024	

Decision on Speech-To-Text problem

Status	COMPLETE
Impact	HIGH
Driver	@yucpeng1
Approver	client
Contributors	@Bowen Fan @yucpeng1 @Tianqi Wang
Informed	client
Due date	Apr 10, 2024
Resources	2024-04-10 Client Meeting notes

Relevant data

Speech-to-Text AI: speech recognition and transcription

https://openai.com/research/whisper

Background

This decision is about the speech-to-text requirement. The client requires the transcript for the video and would like to separate the speaker in the transcript. We tried out 2 ways of STT API, Google Cloud and OpenAI whisper. They have their pros and cons. After meeting with the client, we decided to stick with the whisper.

Options Considered

	Option 1	Option 2
Description	**Using Google Cloud API to do STT task.**	**Using OpenAI Whisper API to do STT task.**
Pros and cons	➕ Can separate speaker. ➖ Can not hear the robot. ➖ Accuracy is low.	➕ Accuracy is high ➕ Can hear the robot clearly ➖ Can not separate speaker
Estimated cost	MEDIUM	MEDIUM

Action items

- [x] Discuss with the client which option they prefer.

Outcome

Our development group will use Whisper API to do the STT task. Meanwhile, we will keep looking for better tools to separate the speaker.

On the other hand, the client provided an idea to polish and enhance the voice of the robot and to see if Google Cloud will work, this will be tested during development.

Decision on Predefined Booklist

Status	IN PROGRESS
Impact	HIGH
Driver	@Yujie Zheng
Approver	
Contributors	
Informed	client
Due date	Apr 23, 2024
Resources	

Background

The client requests a pre-defined booklist for the annotation task. However, as a web app without a database, we have no ability to save data. As a result, the pre-define booklist can either be uploaded or manually imported by the user in the format of special rules(for example CSV).

Options considered

	Option 1 upload and download booklist	**Option 2 manually imported in text editor**
Description	In this option, users are provided with the ability to upload a file containing their predefined annotation booklist. This file can be in a variety of formats, such as CSV, JSON, or XML, which the web app will parse to utilize the contained data. Users can also download their current booklist in one of these formats, which they can edit offline and re-upload at their convenience.	This option involves a text editor integrated into the web application, where users can directly type their annotation booklist using a specific format, such as CSV. This format will allow the app to differentiate between various entries properly. The use of a text editor simplifies direct manipulation of the booklist on the web interface itself without the need for handling files.
Pros and cons	⊕ Users can transfer their booklists between different systems or back them up locally. ⊖ Users are responsible for maintaining their own files, which could lead to issues if files are lost or corrupted.	⊕ Eliminates the need for users to manage physical files, reducing the risk of data loss due to file corruption or misplacement. Users interact with a straightforward text-editing environment, which is easier to use. ⊖ As the size of data grows, the text editor might become cumbersome to manage,

	Handling file uploads and downloads adds additional complexity to the web app's backend, requiring more robust error handling and security measures.	making it harder to navigate and edit extensive lists.
Estimated cost	LARGE	SMALL

✅ Action items

- [ ] Discuss with the client during the meeting on April 23th.

🌟 Outcome

Option 2 is good enough.

Decision on intro page

Status	NOT STARTED / IN PROGRESS / COMPLETE
Impact	HIGH / MEDIUM / LOW
Driver	
Approver	
Contributors	
Informed	
Due date	
Resources	

Relevant data

Background

Options considered

	Option 1	Option 2
Description		
Pros and cons	⊕ ⊖	⊕ ⊖
Estimated cost	LARGE	MEDIUM

Action items

- [ ]

Outcome

Sprint Management with Trello Board

Sprint Task Tracking (Trello Board)

Note:

- *For each task breakdown, please read the ID as:* ***"Epic Id . UId . Taks Id"*** *(where UId is the user story Id. Please refer to the User Story Table)*
- *The Epic assignment is not displayed properly in the Confluence embedded link, please directly visit our Trello board to view the Epic assignment.*

Sprint Tracking

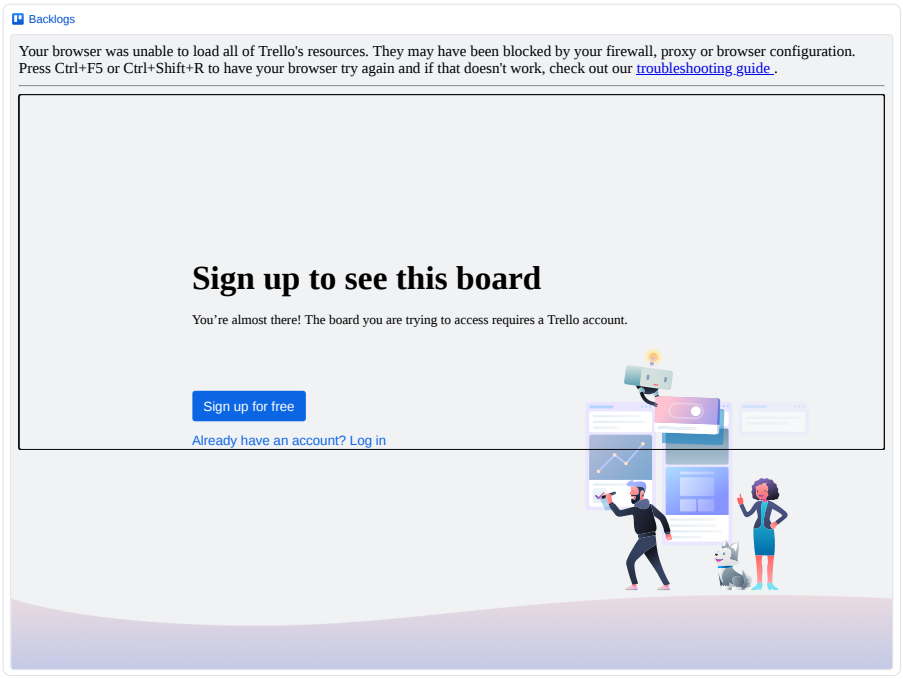
Your browser was unable to load all of Trello's resources. They may have been blocked by your firewall, proxy or browser configuration.
Press Ctrl+F5 or Ctrl+Shift+R to have your browser try again and if that doesn't work, check out our troubleshooting guide .

Sign up to see this board

You're almost there! The board you are trying to access requires a Trello account.

Sign up for free

Already have an account? Log in


Backlogs
Your browser was unable to load all of Trello's resources. They may have been blocked by your firewall, proxy or browser configuration.
Press Ctrl+F5 or Ctrl+Shift+R to have your browser try again and if that doesn't work, check out our troubleshooting guide .
Sign up to see this board
You're almost there! The board you are trying to access requires a Trello account.
Sign up for free
Already have an account? Log in

Sprint 1 Deliverables

Sprint 1

Owned by Tianqi Wang • Updated on Mar 21, 2024

Sprint 1 check-list https://comp90082-2024-na-koala.atlassian.net/wiki/spaces/comp900822/pages/3014657 Sprint 1 task distribution https://comp90082-2024-na-koala.atlassian.net/wiki/spaces/comp900822/pages/6324238 Sprint 1 Retrospective Summary https://comp90082-2024-na-koala.atlassian.net/wiki/spaces/comp900822/pages/6160408

Confluence

Open preview

Sprint 2 Deliverables

Sprint 2

Owned by Yujie Zheng • Updated on May 23, 2024

Sprint 2 Planning https://comp90082-2024-na-koala.atlassian.net/wiki/spaces/comp900822/pages/7864332 Sprint 2 check-list https://comp90082-2024-na-koala.atlassian.net/wiki/spaces/comp900822/pages/16482305 Sprint 2 Retrospective summary https://comp90082-2024-na-koala.atlassian.net/wiki/spaces/comp900822/pages/28835851

Confluence

Open preview

Sprint 1

Sprint 1 check-list

Sprint 1 check-list

Owned by Tianqi Wang • Updated on Mar 21, 2024

Sprint 1 (checklist) This checklist helps you double-check your work for Sprint 1. ## Background description, client goals, motivation [✅] Project overview, background and goals were created. (not mandatory but recommended for all projects - these resources would help with the design sprint and project overview) [✅] DO-BE-FEEL list and GOAL

Confluence

Open preview

Sprint 1 task distribution

Sprint 1 task distribution

Owned by Tianqi Wang • Updated on Mar 19, 2024

User stories, use cases, Personas. @Harry Wang Confluence: @yuchsong2 @yucpeng1 meeting notes Sprint planning/retrospective Daily Standups Project overview, project goal, DO-BE-FEEL list, goal model @Tianqi Wang GitHub structure and Readme. @Bowen Fan Trello board. @Yujie Zheng

Confluence

Open preview

Sprint 1 Retrospective Summary

Sprint 1 Retrospective Summary

Owned by yuchsong2 • Updated on Mar 19, 2024

@Tianqi Wang @Yujie Zheng @Bowen Fan @Harry Wang @yuchsong2 @yucpeng1 @Jinhao He Sprint Number : 1 Duration : 4 March 2024 - 22 March 2024 What went well? Teamwork : Collaboration and support among team members facilitates the smooth running of the work. Task completion was good : the sprint completed all planned tasks and team members performed we

Confluence

Open preview

Sprint 1 Planning: Inception

Sprint 1: Requirement Analysis and Planning Phase

1. **Gather Detailed Requirements and Specifications**
   - Meetings with PhD students for requirements gathering.
   - Identifying user stories based on these requirements.
2. **Survey of Existing Annotation Tools**
   - Research and document features of existing tools (e.g., Elan).
   - Identify innovative features and best practices that can be integrated.

Sprint 1 Tasks

For Sprint 1, the focus is on the initial phase of Requirement Analysis and Planning. Here's a detailed breakdown of tasks:

- **Task 1: Initial Team Meeting**
  - Objective: Align the team on the project's goals and methodologies.
  - Duration: 1 day
- **Task 2: Requirement Gathering Meetings**
  - Objective: Conduct meetings with PhD students to gather detailed requirements.
  - Duration: 1 week
  - Deliverable: A requirements document outlining user needs and expected features.
- **Task 3: Research on Existing Tools**
  - Objective: Survey existing annotation tools and document their features.
  - Duration: 1 week
  - Deliverable: A comparison report highlighting potential features for ROSAnnotator.

Sprint 1 check-list

Sprint 1 (checklist)

This checklist helps you double-check your work for Sprint 1.

Background description, client goals, motivation

- [✅] Project overview, background and goals were created.

(*not mandatory but recommended for all projects - these resources would help with the design sprint and project overview*)

- [✅] DO-BE-FEEL list and GOAL MODEL were created.
- [✅] The goal model is consistent with the client understanding of the problem and with DO-BE-FEEL list.
- [✅] Personas are based on the research done by students and the discussion with industry partners.
- [✅] Personas are inclusive and diverse.

Analysis of requirements (User Stories or Use Cases)

- [✅] The analysis of requirements was performed on most of the existing requirements.
- [✅] The [new set of] requirements is consistent to the scope of the project, completely cover the new capabilities required by the client and are well documented/structured/organised on Confluence.
- [✅] The requirements can be documented in the form of user stories or use cases, supplementary specification of design/implementation/deployment requirements, prototypes, and others. It may also be necessary to be explicit about what is not in scope to define the scope boundary more clearly.
- [✅] We used ChatGPT to generate user stories to our project. On Confluence Space, we documented the prompt we've used, what user stories were generated WITH and WITHOUT ChatGPT.

Development environment

- [✅] Confluence is organised (cover page, project details, requirements, technical details about the project, meeting minutes and so on).
- [✅] Trello (or Github projects or JIRA) is created, structured and organized.
- [✅] Previous/existing project is deployed and could be used/tested as part of this requirements engineering phase.
- [✅] README file is updated and provide details about the project, workflow (branches/naming conventions and so on).

Plan

- [✅] A plan (or discussion on what to do next) was provided (requirements to develop, technologies to use, infrastructure to deploy the project) for Sprint 2 and Sprint 3.
- [✅] Requirements were estimated and prioritised.
- [✅] Backlog items can be found in Trello (or Github project or JIRA).

Meetings

- [✅] Meetings are recorded in Confluence and only. They were NOT exported to Github as they're part of internal process.

GitHub

- [✅] Folders are structured (On Canvas, visit Assignment -> "Sprint 1: Confluence Space, project background and elicitation documents" page: you can find requirements for folders' structure.)

- [✅] Sprint 1 documents were exported from Confluence and added to the repository (and are updated)
- [✅] README file is updated and explain the team's repository
- [✅] A baseline tag was generated for this Sprint (On Canvas, visit Assignment -> "Sprint 1: Confluence Space, project background and elicitation documents" page: you can find requirements for the baseline tag)

Additional Information

do you have any other additional information you'd like to share with us? Please add it here.

Sprint 1 task distribution

1. User stories, use cases, Personas. @Harry Wang
2. Confluence: @yuchsong2 @yucpeng1
    a. meeting notes
    b. Sprint planning/retrospective
    c. Daily Standups
3. Project overview, project goal, DO-BE-FEEL list, goal model @Tianqi Wang
4. GitHub structure and Readme. @Bowen Fan
5. Trello board. @Yujie Zheng

Sprint 1 Retrospective Summary

- @Tianqi Wang
- @Yujie Zheng
- @Bowen Fan
- @Harry Wang
- @yuchsong2
- @yucpeng1
- @Jinhao He

Sprint Number: 1

Duration: 4 March 2024 - 22 March 2024

What went well?

- **Teamwork**: Collaboration and support among team members facilitates the smooth running of the work.
- **Task completion was good**: the sprint completed all planned tasks and team members performed well on the tasks.
- **Active participation in the discussion**: the group members were actively involved in the discussion, presenting different points of view and discussing them.
- **Efficient completion of tasks**: everyone completed their part of the task efficiently.

Areas for improvement

- **Scheduling of meetings**: Meetings were not scheduled appropriately and there were occasional absences.

action plan

Adjust Meeting Times: Re-evaluate and adjust the scheduling of team meetings to reduce absences, and emphasize the meeting time in Slack to ensure that all team members receive the notification.

Summary

The team excelled in communication, task completion and quality control in this sprint. The active participation of members and efficient execution were key to this success. However, the meeting schedule needs to be improved to ensure that all members can attend. In the next sprint, we will adjust the meeting times to better accommodate team members' schedules.

Sprint 2

Sprint 2 Planning

Sprint 2 Planning: Development

Owned by Yujie Zheng • Updated on Apr 30, 2024

Sprint 2 Planning: Development (March 25 - April 26) Overview Sprint 2 will span from March 25 to April 26, focusing on enhancing the core functionality and user interface of our audio-to-transcript application with speaker separation. This period aims to solidify our backend infrastructure, streamline front-end interactions, and explore advanced d

Confluence

Open preview

Sprint 2 check-list

Sprint 2 checklist

Owned by Tianqi Wang • Updated on May 2, 2024

Each group is to submit ONE (and only) checklist here on Canvas. Team-based submission (not individual-based). ✅ ## Confluence (infra) Make sure that your Confluence satisfies the following criteria: [✅] The students have produced an excellent structure of the project on Confluence. [✅] Easy to find contents on pages. [✅] Most of the conte

Confluence

Open preview

Sprint 2 Retrospective summary

Sprint 2 Retrospective

Owned by Tianqi Wang • Updated on Apr 30, 2024

@Tianqi Wang @Yujie Zheng @Bowen Fan @Harry Wang @yuchsong2 @yucpeng1 @ABHISHEK TUMMALAPALLI Sprint Number : 2 Duration : 29 March 2024 - 2 May 2024 What went well? Effective Communication : Clear and open communication among team members enhanced understanding and coordination throughout the project. High-Quality Deliverables : All output met or

Confluence

Open preview

Sprint 2 Planning: Development

Sprint 2 Planning: Development (March 25 - April 26)

Overview

Sprint 2 will span from March 25 to April 26, focusing on enhancing the core functionality and user interface of our audio-to-transcript application with speaker separation. This period aims to solidify our backend infrastructure, streamline front-end interactions, and explore advanced deployment options, ensuring a robust foundation and an intuitive user experience.

Justification

The enhancement of the core functionality and user interface in this sprint is essential for meeting the evolving demands of our target users, who require a robust and intuitive audio-to-transcript application. The introduction of speaker separation is particularly vital for contexts where multiple speakers are present, such as meetings, interviews, and conferences, ensuring that the transcription accurately reflects who said what. This feature not only enhances user satisfaction but also increases the usability of the application in professional and academic settings.

Moreover, the development of a comprehensive architecture diagram and the careful planning of backend and frontend integrations are designed to minimize potential technical debt and streamline future development efforts. This strategic approach allows for more predictable scaling and maintenance of the application as user demand grows. Early focus on these areas will prevent costly reworks and ensure that the product can efficiently handle increased workloads as it matures.

The exploration of advanced deployment options like ROS in Docker containers highlights our commitment to leveraging cutting-edge technology to enhance our application’s capabilities. While not immediately critical, this investigation is an investment in future-proofing the application and exploring possibilities that could set our product apart from competitors.

Objectives and Key Deliverables

1. **Audio to Transcript with Speaker Separation**: Prioritize the development of a reliable and accurate system for converting audio to text while identifying and separating speakers. This critical functionality forms the backbone of our application, necessitating an early sprint focus to allow ample time for integration and testing.
2. **Architecture Diagram Creation**: Early establishment of a comprehensive architecture diagram is crucial. It will guide the development process, ensuring all team members understand the system's structure, components, and how they interact. This clarity will facilitate more efficient development and integration across all parts of the project.
3. **Backend Integration for Extracted Code**: Organize the backend to handle audio processing and transcript generation efficiently. This involves creating a scalable and secure architecture that can manage heavy workloads and ensure smooth communication with the front end.
4. **Frontend Prototype Design**: Designing an intuitive and functional frontend prototype is essential for user interaction. This task will follow the backend setup to align the design with the backend capabilities, focusing on displaying the transcript effectively and allowing users to navigate the audio and text seamlessly.
5. **Transcript Scrolling Display with Timeline Jump**: Implement a user-friendly feature for scrolling through the transcript and jumping to specific video timelines. This feature enhances the user experience by providing efficient navigation and interaction with the content.
6. **Video Player Subtitle Toggle**: Add a feature in the video player to toggle subtitles on and off, generated from the transcript. This functionality must be user-centric, offering flexibility in how users engage with the video content.
7. **Exploration of ROS Docker**: Investigate the implementation of ROS (Robot Operating System) within Docker containers. This exploration, while not immediately critical, could provide valuable insights into improving processing capabilities or simulation environments in the future.
8. **Story Point Replan**: Midway through the sprint, reassess and replan story points for tasks based on progress and any unforeseen challenges. This adjustment ensures that the team remains on track and that priorities are correctly aligned with project goals.

Sprint Schedule and Milestones

- **Week 1 (March 25 - March 31)**: Kick-off with audio to transcript development and start architecture diagram. Research speech recognition and speaker diarization technologies.
- **Week 2 (April 1 - April 7)**: Continue backend integration and finalize architecture diagram. Begin frontend prototype design.
- **Week 3 (April 8 - April 14)**: Complete frontend design and start implementation. Initiate transcript scrolling and video subtitle toggle features.
- **Week 4 (April 15 - April 21)**: Conduct ROS Docker exploration. Implement and integrate backend and frontend functionalities. Start comprehensive testing of all features.
- **Week 5 (April 22 - April 26)**: Final adjustments, additional testing, and story point replan. Prepare for end-of-sprint review, focusing on lessons learned and planning for the next development phase.
- **Week 6 (April 29 - May 2): Documentation and Closure:** Extra week for documentation. Detailed documentation of all features developed during the sprint, updates to the architecture diagram, and preparation of end-of-sprint materials.

Sprint 2 checklist

Each group is to submit ONE (and only) checklist here on Canvas. Team-based submission (not individual-based).

- ✅

Confluence (infra)

Make sure that your Confluence satisfies the following criteria:

- [✅] The students have produced an excellent structure of the project on Confluence.
- [✅] Easy to find contents on pages.
- [✅] Most of the contents are visible and editable (no unnecessary attachments).

Confluence (contents and consistency)

Make sure that your Confluence satisfies the following criteria:

- [✅] Contents are available and updated on Confluence (meeting minutes, scope of the project, diagrams, technologies used in the project, user stories, test cases).
- [✅] Contents are consistent with trello (or github project) and with their code repositories.

Task Tracking

Make sure that your task tracking satisfy the following criteria:

- [✅] Students organized a product backlog and a lower-level sprint backlog.
- [✅] Tasks in the sprint backlog were estimated, have an appropriate due date and have a sufficiently low level of granularity.
- [✅] Tasks are also clear, linked to their user stories (Confluence) and offer additional description when necessary.
- [✅] A link to your Trello or JIRA (Anyone with link can access that resource) was made available to your marker.

Code Review (Sprint 2)

Make sure that your code review satisfy the following criteria:

- [ ] Students documented their peer-to-peer or chatgpt code review on GitHub (pull request comment)[THIS ITEM IS OPTIONAL].
- [✅] Students documented their peer-to-peer or chatgpt code review on Confluence (new page on Confluence to document how you performed your code review - who participated in that, when did that happen, number of issues identified and so on).
- [ ] In case you used ChatGPT in this sprint, please disconnect your GitHub repository from our ChatGPT Code Review one so you don't get charged in the future (or, make sure you continue to use this carefully and only when strictly necessary - do not use it for all commits)

Sprint Planning and Review

Make sure that your sprint planning and review are documented on Confluence:

- [✅] Clear indication that sprint planning was followed this sprint and a clear, consistent, updated sprint planning for Sprint 3.
- [✅] Clear indication that sprint review was followed this sprint. Team organised a meeting for this, documented discussions, reflections and next steps to be taken in Sprint 3.

Ethical Considerations

Make sure that your ethical considerations are documented on Confluence:

- [✅] Clear indication that students discussed/reflected on project ethical issues.

Cyber Security

Make sure that your cyber security considerations are documented on Confluence:

- [✅] Clear indication that students discussed/reflected on project cyber security issues.

Product

Make sure that your product satisfy the following criteria:

- [] Product is deployed and an URL is available on Confluence and Github README so client can access current version of this software.
- [✅] FOR PROJECTS YOU CANT DEPLOY IT NOW: Can you emulate the project and demonstrate current progress for us in a short recorded video? we need to be able to measure your development progress in Sprint 2, that's all.

Meetings

Make sure your meetings (team meetings, supervision meetings and meetings with industry partners) are documented in Confluence (and only).

- [✅] Meetings are recorded in Confluence and only. They were NOT exported to Github as they're part of an internal process.

GitHub

Make sure that:

- [✅] Folders are structured.
- [✅] Sprint 2 documents were exported from Confluence and added to the repository (and are updated)
- [✅] README file is updated and explain the team's repository and new release
- [✅] A baseline tag was generated for this Sprint

Additional Information

do you have any other additional information you'd like to share with us? Please add it here.

Sprint 2 Retrospective

- @Tianqi Wang
- @Yujie Zheng
- @Bowen Fan
- @Harry Wang
- @yuchsong2
- @yucpeng1
- @ABHISHEK TUMMALAPALLI

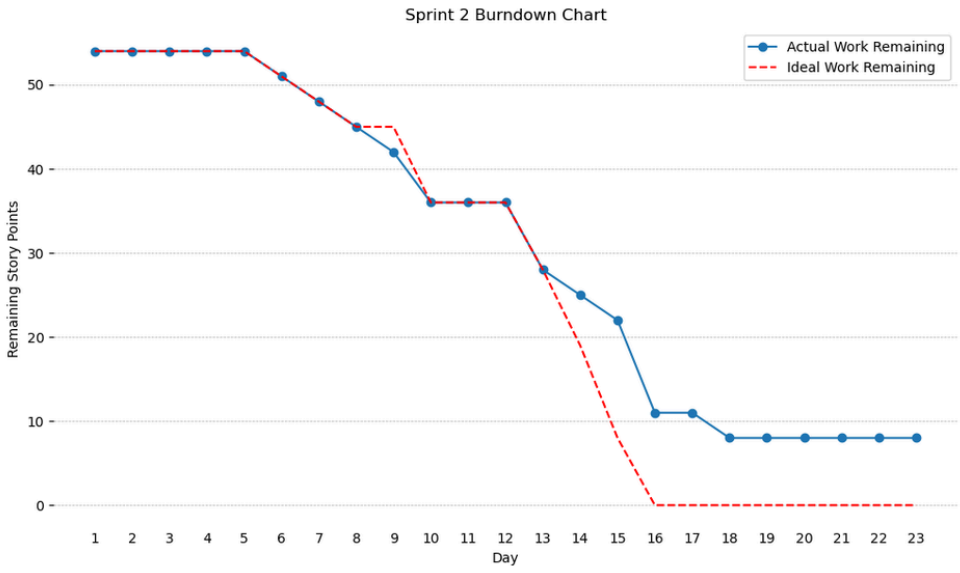
Sprint Number: 2

Duration: 29 March 2024 - 2 May 2024

What went well?

- **Effective Communication**: Clear and open communication among team members enhanced understanding and coordination throughout the project.
- **High-Quality Deliverables**: All output met or exceeded the expected standards, reflecting the team's dedication and expertise.
- **Engagement and Enthusiasm**: Team members showed high levels of engagement and enthusiasm, contributing positively to the team atmosphere.
- **Problem Solving**: Team members effectively addressed and resolved challenges as they arose, maintaining project momentum.

Sprint burndown chart



Areas for improvement

- **Task Completion Timeliness**: There were delays in completing some tasks, which impacted the overall project timeline.
- **Unfinished Tasks**: A few tasks remained incomplete by the end of the sprint, needing reassignment or additional resources to ensure completion.
- **Feedback and Revisions**: The process for gathering feedback and implementing revisions could be streamlined to accelerate project advancement.
- **Resource Allocation**: There were occasional mismatches in resource allocation, which affected the efficiency of task execution.

Action plan

- **Enhanced Project Tracking**: Implement a more robust project management tool to monitor task progress and deadlines more effectively. Ensure that all team members are trained on how to use this tool efficiently to keep tasks on track and address delays promptly.

Summary

During the second sprint, our team excelled in communication, quality of deliverables, engagement, and problem-solving, which were key contributors to the project's overall progress. However, we faced challenges with task completion timeliness, incomplete tasks, and inefficient resource allocation, as indicated by our burndown chart. Moving forward, we plan to implement a robust project management tool to enhance tracking and improve task completion rates, aiming to streamline our processes for better efficiency and output in future sprints.

Sprint 3

Sprint 3 Planning

Sprint 3 Planning: Development

Owned by Yujie Zheng • Updated on May 23, 2024

Sprint 3 Planning: Development（April 27 - May 24) Objective: Focus on refinement, and addressing any identified issues or improvements. Duration: 4 weeks Integration with Auto-Transcription Integrate auto-transcription service for speech-to-text conversion. Develop annotation functionalities. Duration: 1 Week Estimation: After voting and discussion

Confluence Open preview

Sprint 3 check-list

Sprint 3 Checklist

Owned by Tianqi Wang • Updated on May 23, 2024

Each group to submit ONE (and only) checklist here on Canvas. ## Confluence (infra) Make sure that your Confluence satisfies the following criteria: [✅] The students have produced an excellent structure of the project on Confluence. [✅] Easy to find contents on pages. [✅] Most of the contents are visible and editable (no unnecessary attachme

Confluence Open preview

Sprint 3 Retrospective summary

Sprint 3 Retrospective

Owned by yucpeng1 • Updated on May 23, 2024

@Tianqi Wang @Yujie Zheng @Bowen Fan @Harry Wang @yuchsong2 @yucpeng1 @ABHISHEK TUMMALAPALLI Sprint Number : 3 Duration : 6 May 2024 - 22 May 2024 What went well? Effective Bug Resolution : Our team tackled a significant number of bugs throughout the development process. Each bug fix contributed to the stability and reliability of the final produ

Confluence Open preview

Sprint 3 Demo video

https://drive.google.com/file/d/1TVW8IRGzAULBHdEeJwETAcFMruCib3e5/view?usp=sharing Connect your Google account Open with unimelb email account.

Sprint 3 Planning: Development

Sprint 3 Planning: Development（April 27 - May 24)

Objective: Focus on refinement, and addressing any identified issues or improvements.

Duration: 4 weeks

1. **Integration with Auto-Transcription**
   - Integrate auto-transcription service for speech-to-text conversion.
   - Develop annotation functionalities.
   - Duration: 1 Week
   - Estimation: After voting and discussion, this part is given 5 story points.
2. **Annotation Features**
   - Develop custom scales and annotation tools for users.
   - Duration: 2 Week
   - Estimation: After voting and discussion, this part is given 11 story points.
3. **Automatic Annotation Exploration**
   - Research and prototype automatic annotation methods.
   - Duration: Out-of-scope
   - Estimation: After voting and discussion, this part is given 0 story points. Since it is out-of-scope
4. **Front-End Refinement**
   - Add “login page“
   - Gather feedback on the front-end design from users or stakeholders.
   - Implement improvements and refinements to enhance user experience and usability.
   - Duration: Throughout Sprint 3
   - Estimation: After voting and discussion, this part is given 5 story points.
5. **Testing**
   - Conduct thorough testing, including unit tests, integration tests, and user acceptance tests.
   - Address any issues or bugs identified during testing.
   - Duration: 1 Week

Sprint Velocity Estimation

- 🎯 Introduction
- ✏️ Story Point Assignment Approach
  - Base Units
  - Sequence for Larger Stories
  - Breakdown into Subtasks
  - Adjustment During Sprint
  - Planning Poker Method
- 📊 Backlogs
  - Produce Backlog
  - Sprint Task Tracking
- 🗒️ Agenda for the Meeting
- 🔢 Story Points Assignment Voting Record

🎯 Introduction

This page documents the Velocity Estimation part for the Sprint 3 Planning meeting. The primary objective of this meeting is to estimate the story points for our user stories assigned to sprint 3 and plan the tasks accordingly. Our approach to story point assignment helps manage workload, adjust velocity, and track the completion of points by each developer effectively.

> Note: For a detailed justification on this topic, please visit our User Stories page.

✏️ Story Point Assignment Approach

We assign story points to each user story primarily based on the difficulty level of implementation. Here's how we approach it:

Base Units

- **3 Points:** Represents a small task with low complexity.
- **5 Points:** Represents a moderate task with medium complexity.

Sequence for Larger Stories

We estimate larger stories by adding repetitions of these base units together. The sequence follows:

- 3, 5, 6, 8, 11, 13, 15, 16, etc.
- Example: 16 = 3 x 2 + 5 x 2

Note: No single story is assigned more than 16 points. If necessary, we break it down into smaller stories.

Breakdown into Subtasks

Each story is further divided into smaller, actionable tasks assigned to individual team members during a sprint. The total story points for a user story are the sum of all the story points of its subtasks.

Adjustment During Sprint

For ongoing tasks within a sprint, if we find that the initial estimation significantly deviates from the actual workload, we adjust the story points accordingly.

Planning Poker Method

For future sprints, we estimate the number of subtasks for each story and use the Planning Poker method to estimate the points for each task according to their difficulties during the planning meeting. The estimation that receives the highest vote will be assigned.

📊 Backlogs

Produce Backlog

Please visit our Product Backlog list on Trello to view the user stories being selected for this sprint and story points distributions.

Sprint Task Tracking

Please visit our Sprint Task Tracking Board on Trello to see the task breakdown.

🗒️ Agenda for the Meeting

1. **Review Previous Sprint:**
   - Discuss completed tasks and carry-overs.
   - Review any deviations in initial estimations and adjustments made.
2. **New User Stories & Task Breakdown :**
   - Present and discuss new user stories from the Product Backlog.
   - Break down stories into subtasks.
3. **Story Point Estimation:**
   - Use the Planning Poker method for each subtask.
   - Assign story points based on team consensus.
4. **Task Assignment:**
   - Assign tasks to individual team members.
   - Ensure balanced workload distribution.

🔢 Story Points Assignment Voting Record

During our meeting, we will record the voting process for assigning story points to each user story. Below is the format for recording the voting process:

User Story ID	Description	Subtasks	Participants	Votes (Story Points)	Final Story Points
Loading Data					
1.3	Handle large volumes of ROSBag data efficiently, ensuring quick	2	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng	3: 1 votes 6: 5 votes	6

	loading and parsing times		@yuchsong2 @yucpeng1		
3.2	Easily save and export annotated datasets in a common format (e.g., ROSBag)	2	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	3: 2 votes 6: 4 votes	6
5.1	Manage and switch between multiple ROSBags easily within the application	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	5: 2 votes 8: 4 votes	8
Interface Development					
3.1	Intuitive and interactive interface for annotating video streams ~~and 3D point clouds~~	2	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	6: 1 votes 8: 2 votes 11: 3 votes	11
Annotation Features					
3.3	Manually annotate the data(video and audio) on a timeline with detailed labels using my domain knowledge, such as "user confusion", "successful interaction", or "navigation issue"	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	3: 2 votes 5: 4 votes	5
3.4	Easily edit or delete annotations	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	5: 6 votes	5
3.5	Annotate the transcribed text with custom tags or notes	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	5: 1 votes 6: 5 votes	6
3.8	Store custom annotation sets within the system	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	3: 2 votes 6: 4 votes	6
3.9	Synchronize annotations across different data streams (video, audio transcripts, 3D point clouds)	2	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	6: 1 votes 8: 4 votes 11: 1 votes	8
6.3	Customize rules for automatic annotation	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	3: 2 votes 6: 4 votes	6
Transcribed Features					
4.2	Correlate transcribed text with specific moments and events in the video and 3D point cloud data	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	6: 4 votes 8: 2 votes	6
DevOps					
7.1	The App can be run on any platform without heavy manual configuration	3	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	11: 1 votes 13: 2 votes	15

				15 : 3 votes	
7.2	Start using the app with only a few simple commands after pull down from GitHub	1	@Tianqi Wang @Bowen Fan @Harry Wang @Yujie Zheng @yuchsong2 @yucpeng1	8 : 1 votes 11 : 4 votes 13 : 1 votes	11

Sprint 3 Retrospective

- @Tianqi Wang
- @Yujie Zheng
- @Bowen Fan
- @Harry Wang
- @yuchsong2
- @yucpeng1
- @ABHISHEK TUMMALAPALLI

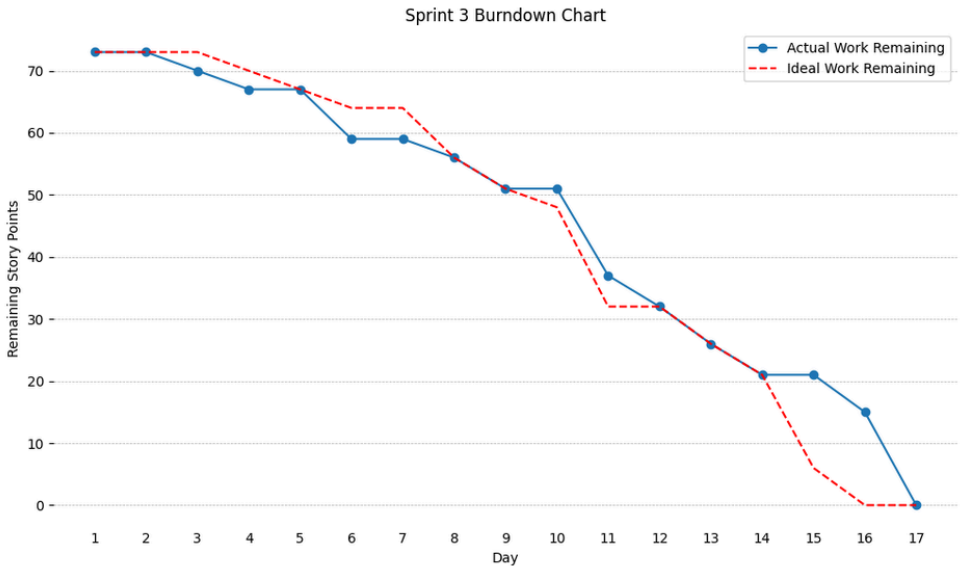
Sprint Number: 3

Duration: 6 May 2024 - 22 May 2024

What went well?

1. **Effective Bug Resolution**: Our team tackled a significant number of bugs throughout the development process. Each bug fix contributed to the stability and reliability of the final product, ensuring a seamless user experience.
2. **Successful Feature Implementation**: We achieved the successful implementation of numerous new features. These enhancements not only expanded the functionality of our system but also addressed user needs more comprehensively.
3. **UI Beautification**: A major focus was placed on improving the user interface. The newly designed UI is more visually appealing and user-friendly, leading to an improved overall user experience.

Sprint burndown chart



Areas for improvement

- **Unfinished Features**: We were unable to implement certain planned features. The extensive bug-fixing efforts required additional time and resources, leaving some functionalities incomplete by the end of the sprint.
- **Time Management for Bug Fixes**: While resolving bugs was essential, the time spent on these fixes impacted the development of new features. Balancing bug resolution with feature development more effectively could enhance overall project outcomes.
- **Documentation Practices**: Improving documentation practices could help team members quickly understand and address issues, reducing the time needed for onboarding and troubleshooting.
- **Team Coordination**: Enhancing coordination between different teams or departments involved in the project could lead to more synchronized efforts and prevent overlaps or gaps in responsibilities.

Action plan

- **Feature Implementation Strategy**: Develop a clearer strategy for feature implementation that includes dedicated time blocks for both bug fixes and new feature development. This will ensure that neither task is neglected and that planned features are more likely to be completed within the sprint.

Summary

Our team effectively resolved a significant number of bugs, implemented numerous new features, and enhanced the user interface, leading to a more stable and user-friendly product. However, we faced challenges with unfinished features due to extensive bug-fixing efforts, impacting new feature development. To improve, we plan to balance bug resolution with feature development, enhance documentation practices, and improve team coordination to ensure timely task completion and streamlined project execution.

Sprint 3 Checklist

Each group to submit ONE (and only) checklist here on Canvas.

Confluence (infra)

Make sure that your Confluence satisfies the following criteria:

- [✅] The students have produced an excellent structure of the project on Confluence.
- [✅] Easy to find contents on pages.
- [✅] Most of the contents are visible and editable (no unnecessary attachments).

Confluence (contents and consistency)

Make sure that your Confluence satisfies the following criteria:

- [✅] Contents are available and updated on Confluence (meeting minutes, scope of the project, diagrams, technologies used in the project, user stories, test cases).
- [✅] Contents are consistent with trello (or github project) and with their code repositories.

Task Tracking

Make sure that your task tracking satisfy the following criteria:

- [✅] Students organised a product backlog and a lower-level sprint backlog.
- [✅] Tasks in the sprint backlog were estimated, have an appropriate due date and have a sufficiently low level of granularity.
- [✅] Tasks are also clear, linked to their user stories (Confluence) and offer additional description when necessary.
- [✅] A link to your Trello or JIRA (Anyone with link can access that resource) was made available to your marker.

Code Review (Sprint 3)

Make sure that your code review satisfy the following criteria:

- [✅] Students documented their automated code review on Confluence (new page on Confluence to document how you performed your code review - who participated in that, when did that happen, number of issues identified and so on). Example of contents to include in this page: how many times did you perform code reviews? what was the selection criteria to decide what codes to be inspected? once code was reviewed, what happened next? how did you and your team review the received feedback from AI? provide examples on reviewed codes, received feedback and what actions happened after that (what changed in the code, what didn't change, and why). Comprehensive documentation here.
- [✅] In case you didn't use our OpenAI Key in this sprint (you created your own), please disconnect your GitHub repository from our ChatGPT Code Review one so you don't get charged in the future (or, make sure you continue to use this carefully and only when strictly necessary - do not use it for all commits)

Sprint Planning and Review

Make sure that your sprint planning and review are documented on Confluence:

- [✅] Clear indication that sprint planning was followed this sprint and a clear, consistent, updated sprint planning for Sprint 4.
- [✅] Clear indication that sprint review was followed this sprint. Team organised a meeting for this, documented discussions, reflections and next steps to be taken in Sprint 4.

Ethical Considerations

Make sure that your ethical considerations are documented on Confluence:

- [✅] Clear indication that students discussed/reflected on project ethical issues.

Cyber Security

Make sure that your cyber security considerations are documented on Confluence:

- [✅] Clear indication that students discussed/reflected on project cyber security issues.

Product

Make sure that your product satisfy the following criteria:

- [ ] Product is deployed and an URL is available on Confluence and Github README so client can access current version of this software.
- [✅] FOR PROJECTS YOU CANT DEPLOY IT NOW: Can you emulate the project and demonstrate current progress for us in a short recorded video? we need to be able to measure your development progress in Sprint 3, that's all.

Meetings

Make sure your meetings (team meetings, supervision meetings and meetings with industry partners) are documented in Confluence (and only).

- [✅] Meetings are recorded in Confluence and only. They were NOT exported to Github as they're part of internal process.

GitHub

Make sure that:

- [✅] Folders are structured.
- [✅] Sprint 3 documents were exported from Confluence and added to the repository (and are updated)
- [✅] README file is updated and explain the team's repository and new release
- [✅] A baseline tag was generated for this Sprint

Additional Information

do you have any other additional information you'd like to share with us? Please add it here.

Sprint 4

Sprint 4 Planning

Sprint 4 Planning: Project Finalization and Delivery

Owned by yuchsong2 • Updated on May 23, 2024

Sprint 4 Planning: Project Finalization and Delivery (May 25 - Jun 7) Objective : Sprint 4 focuses on completing the project, ensuring thorough testing, and delivering a polished version to the client. Duration : 2 weeks Tasks: 1.Release Tag Generation: Create a detailed release tag on Github with all project resources, including documents and diag

Confluence

Open preview

Sprint 4 Checklist

Sprint 4 Checklist

Owned by Tianqi Wang • Updated 2 minutes ago

Meetings Make sure your meetings (team meetings, supervision meetings and meetings with industry partners) are documented in Confluence (and only). [✅] Meetings are recorded in Confluence and only. They were NOT exported to Github as they're part of internal process. ## Release TAG Make sure that: [✅] Students generated a release TAG on Git

Confluence

Open preview

Sprint 4 Retrospective

Sprint 4 Retrospective

Owned by Tianqi Wang • Updated yesterday

@Tianqi Wang @Yujie Zheng @Bowen Fan @Harry Wang @yuchsong2 @yucpeng1 @ABHISHEK TUMMALAPALLI Sprint Number : 4 Duration : 27 May 2024 - 7 June 2024 What is done? Meetings Meetings have been recorded in Confluence and only. They were NOT exported to GitHub as they're part of the internal process. Release TAG A release TAG has been generated on Git

Confluence

Open preview

Final demo video

NA-Koala - Google Drive

drive.google.com

Open preview

Sprint 4 Planning: Project Finalization and Delivery

Sprint 4 Planning: Project Finalization and Delivery (May 25 - Jun 7)

Objective: Sprint 4 focuses on completing the project, ensuring thorough testing, and delivering a polished version to the client.

Duration: 2 weeks

Tasks:

1.Release Tag Generation:

Create a detailed release tag on Github with all project resources, including documents and diagrams from Confluence. This ensures efficient version control and easy access to project materials.

Duration: Throughout Sprint 4

2.Demo Video Creation:

Develop a concise 3-5 minute demo video showcasing the project's key features, functionality, and benefits. The video will serve as a powerful tool for demonstrating the project's capabilities.

Duration: Throughout Sprint 4

3.Project Completion and Deployment:

Finalize the project and deploy it for client use. Package the latest software version into a ZIP file and deliver it to the client for access to the completed product.

Duration: Throughout Sprint 4

4.Presentation Slides Update:

Update the final presentation slides with any new information or insights gained during project development. These slides will communicate essential project details and outcomes to stakeholders and the client.

Duration: Throughout Sprint 4

5.Github Readme Update:

Update the Github README file with a new URL for easy access to the latest software version. This update ensures users and stakeholders can quickly find and access the most recent project version.

Duration: Throughout Sprint 4

Sprint 4 Checklist

Meetings
Make sure your meetings (team meetings, supervision meetings and meetings with industry partners) are documented in Confluence (and only).

- [✅] Meetings are recorded in Confluence and only. They were NOT exported to Github as they're part of internal process.

Release TAG
Make sure that:

- [✅] Students generated a release TAG on Github (containing all project resources, including exported documents/diagrams from Confluence).

Demonstration video of your product
Make sure that:

- [✅] Students generated a 3-5mins (max) video demonstrating their product (examples: CIS Projects)
- [✅] Students uploaded their demo to Confluence

ZIP File
Make sure that:

- [✅] Students created an organised release to the client on Github, including: documents, tests, data samples, prototypes, and images.
- [✅] Release was downloaded from Github, packed in a ZIP file, sent to the client and added to Confluence (under Handover page).

Final Presentation Slides
Make sure that:

- [✅] Added to Confluence and Github. Industry partner will receive it as part of final release package.

Product
Make sure that your product satisfy the following criteria:

- [✅] Product is deployed and an URL is available on Github README so client can access current version of this software (IF APPLIED. If not, please explain).
    - Our product **does not require deployment** because **it is designed to run locally** as a web application within a Docker container. This setup ensures that all necessary components and dependencies are encapsulated within the container, providing a consistent and reproducible environment for the application. Users can easily run the web application on their local machines by following the provided instructions, which involve running the Docker container and accessing the application via `localhost`. This approach simplifies the setup process, avoids potential deployment issues, and ensures that the application performs as expected in a controlled environment. **This arrangement has been discussed and agreed upon with the client, who is aware and supportive of this local deployment method.**

Sprint 4 Retrospective

- @Tianqi Wang
- @Yujie Zheng
- @Bowen Fan
- @Harry Wang
- @yuchsong2
- @yucpeng1
- @ABHISHEK TUMMALAPALLI

Sprint Number: 4

Duration: 27 May 2024 - 7 June 2024

What is done?

Meetings

- Meetings have been recorded in Confluence and only. They were NOT exported to GitHub as they're part of the internal process.

Release TAG

- A release TAG has been generated on GitHub, containing all project resources, including exported documents and diagrams from Confluence.

Demonstration Video of Your Product

- A 3-5 minute demonstration video demonstrating the product has been created.
- The demo video has been uploaded to Confluence.

ZIP File

- An organized release for the client has been created on GitHub, including documents, tests, data samples, prototypes, and images.
- The release was downloaded from GitHub, packed in a ZIP file, sent to the client, and added to Confluence (under the Handover page).

Final Presentation Slides

- Final presentation slides have been added to Confluence and GitHub. The industry partner will receive them as part of the final release package.

Summary

In Sprint 4, we ensured that all team meetings, supervision meetings, and meetings with industry partners were meticulously documented in Confluence, adhering to our internal process without exporting them to GitHub. A comprehensive release TAG was created on GitHub, including all necessary project resources, such as documents and diagrams from Confluence. We produced a concise 4-minute demonstration video showcasing our product and uploaded it to Confluence. An organized release for the client, encompassing documents,

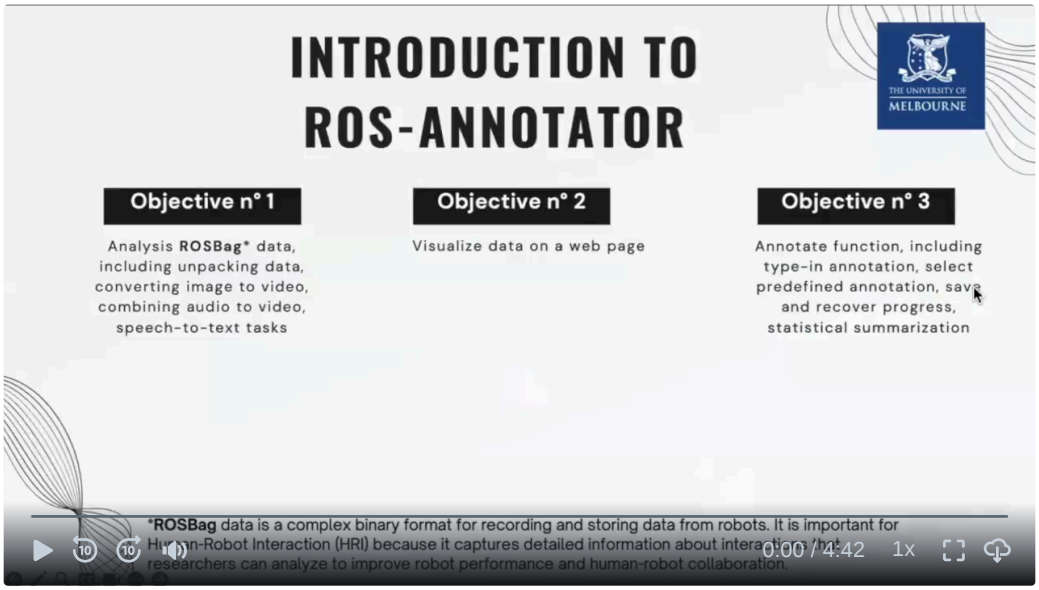
tests, data samples, prototypes, and images, was created on GitHub, zipped, sent to the client, and added to Confluence under the Handover page. The final presentation slides were uploaded to both Confluence and GitHub, ensuring they are included in the final release package for our industry partner.

Reports and Presentation Slide

Presentation Slide

NA-Koala final … ide.pdf
04 Jun 2024, 07:02 AM

Demo video



Cybersecurity Considerations

The cybersecurity considerations for the ROSAnnotator project are designed to ensure the integrity, confidentiality, and availability of data within our system. This report details the specific implementations and protocols we have instituted as part of our cybersecurity strategy, with a focus on how these measures specifically address the identified risks.

Project-Specific Threat Model

The development of a detailed, project-specific threat model is crucial for understanding and mitigating potential security risks associated with the ROSAnnotator project. This model helps us to identify specific threats, assess their potential impact and likelihood, and implement appropriate mitigation strategies.

Threat Model Table:

Threat	Description	Impact	Likelihood	Mitigation Strategy
Unauthorized Data Access	Unauthorized access to sensitive video and audio data	High	Medium	Implement robust access controls and MFA
Data Corruption	Accidental or malicious alteration of data	Medium	Low	Use checksums and data integrity checks
Privacy Breaches	Exposure of personal information from datasets	High	Medium	Data anonymization and use of encryption
Injection Attacks	SQL injection, XSS, etc., compromising system integrity	High	High	Employ secure coding practices and input validation

Role-Based Access Control (RBAC) Detailed Implementation

In response to the identified threats, particularly unauthorized data access and privacy breaches, we have implemented an RBAC system designed to control access based on the sensitivity of the data and the necessity of the role. The RBAC settings are configured as follows:

- **Administrators**: Full control over the system, including access to all data logs, system settings, and user management functionalities. They can also manage encryption keys and perform system audits.
- **Analysts**: Access is limited to viewing and annotating ROSBag data. Analysts cannot modify system settings or view administrative logs to ensure they interact only with data relevant to their tasks.
- **Guests**: Can only view finalized data outputs. They have no access to raw data or the ability to make any annotations, ensuring that sensitive data remains protected.

This precise definition of roles ensures that individuals only have access to data and functionalities essential to their job responsibilities, effectively mitigating the risk of data leaks and unauthorized access.

Secure Design Documentation

The secure design of the ROSAnnotator project is documented comprehensively to outline how security considerations are integrated into the architecture. This documentation includes:

- **Architecture Diagrams**: Visual diagrams that detail the secure architecture of the system, highlighting critical security controls at each layer of the application.
- **Secure Coding Practices**: A list of secure coding practices adopted by the development team, such as input validation, use of parameterized queries, and comprehensive error handling to prevent common vulnerabilities like SQL injection and cross-site scripting (XSS).

Data Protection Strategies

The following strategies have been implemented to protect the integrity, confidentiality, and availability of data within the ROSAnnotator project:

- **Encryption Protocols**: Specifications of encryption protocols used for securing data at rest and in transit, such as TLS for data transmission and AES for data storage.
- **Access Control Policies**: Detailed descriptions of role-based access control (RBAC) policies that ensure only authorized personnel can access specific types of data.
- **Data Integrity Mechanisms**: Implementation of mechanisms like checksums and digital signatures to maintain and verify data integrity.

Conclusion

The cybersecurity enhancements outlined in this report are tailored specifically to the needs of the ROSAnnotator project. By implementing these targeted strategies, we not only enhance the security of the system but also build trust with our users and stakeholders. Regular updates and revisions to our security documentation on Confluence will continue to reflect the evolving nature of our project and the cybersecurity landscape.

Ethical Considerations

The ROSAnnotator project, designed to enhance the analysis of ROSBag data for Human-Robot Interaction (HRI), presents unique ethical challenges that must be addressed to ensure the responsible handling of sensitive data, equitable access, and unbiased analysis. This report outlines the specific ethical issues identified within the project, alongside detailed strategies for their resolution and ongoing management.

Specific Ethical Issues

Privacy and Confidentiality: The project processes potentially sensitive data, including video and audio recordings from HRI environments. Such data can include personal identifiers or sensitive personal interactions, raising significant privacy concerns.

Bias in Automated Systems: The application's reliance on auto-transcription tools and facial recognition technologies poses risks of bias, potentially skewing the analysis of HRI data if the underlying models are not representative of diverse populations.

Informed Consent: Ensuring that individuals whose data is captured and analyzed have provided informed consent is critical, especially given the complex environments in which HRI typically occurs.

Reliability and Accountability: The integrity of data processing and annotation within ROSAnnotator is vital. Errors in data interpretation could misrepresent human behaviors, impacting HRI research and applications.

Access and Equity: It is essential that ROSAnnotator is accessible to all potential users, including those with disabilities, and that it does not create or perpetuate inequalities among user groups.

Resolution Strategies

Enhanced Data Protection Measures: We employ AES-256 encryption for all data at rest and TLS 1.3 for data in transit, ensuring that even in the event of a data breach, the information remains secure. We also implement data anonymization techniques, such as pseudonymization of personal identifiers before storage and processing. This not only complies with GDPR requirements but also minimizes the risk of personal data exposure.

Bias Mitigation: We regularly update our auto-transcription and facial recognition algorithms to address potential biases. Our models are retrained semi-annually on newly collected diverse datasets that represent a wide range of demographics. Fairness audits are conducted following each update to assess and rectify any emergent bias, ensuring the algorithms perform equitably across all user groups.

Robust Consent Management: Our consent management framework includes an easy-to-navigate interface where users can opt-in or withdraw their consent at any time. Upon first use, users are prompted with a detailed consent form that outlines how their data will be used, stored, and protected. Users can revisit and adjust their consent preferences through their profile settings, reflecting changes in real-time.

Accountability Mechanisms: Our system logs all data processing activities, providing an audit trail that can be reviewed in case of discrepancies. If a data processing error is detected, our system triggers an alert, and the issue is logged for review. The development team addresses these issues in a dedicated bi-weekly review session, ensuring accountability and swift resolution.

Ensuring Inclusivity: We adhere to WCAG 2.1 AA standards to ensure web accessibility. Our interface includes features such as text-to-speech, high-contrast visuals, and keyboard navigation options to accommodate users with different abilities. Regular accessibility audits are conducted to identify and rectify any new issues that might prevent users from fully accessing our services.

Team Commitments and Documentation

The development team is committed to regular ethical training to reinforce the importance of ethical design, privacy protection, and unbiased development. Ethical considerations, data handling protocols, and resolution strategies are meticulously documented on Confluence, which includes version-controlled records of all ethical decisions, data handling protocols, and the outcomes of ethical audits. This documentation is updated with each sprint and reviewed in sprint retrospectives, allowing for ongoing assessment and continuous improvement of our ethical practices.

Conclusion

The ethical framework established for the ROSAnnotator project ensures that it not only advances HRI research but does so with a firm commitment to upholding high ethical standards. By addressing each ethical challenge with clear strategies and maintaining rigorous documentation and team training, the project sets a standard for responsibility and inclusivity in technological development.

Presentation Preparation

Presentation Outline

1. Introduction and Project Overview (1 minute)

- **Speakers:** @Tianqi Wang
- **Content:**
  - Brief introduction of the project team members.
  - Overview of the project title: "ROSAnnotator: A Web Application for ROSBag Data Analysis in Human-Robot Interaction."
  - Purpose of the presentation.

2. Architecture + Backend (4 minutes)

- **Speakers:** @Harry Wang
- **Content:**
  - Docker
  - Backend architecture
  - ROSBAG analysis + Speech to text
  - Front page

2. Front-end demonstration (4 minutes)

- **Speakers:** @yuchsong2 @Bowen Fan
- **Content:**
  - Provide an in-depth live demo or video demonstration of the web application showcasing key features:
    - Loading and parsing ROSBag data.
    - Auto-transcription and manual annotation process.
    - Interactive dashboard functionalities.
    - Comparative and composite analysis using multiple ROSBags.

3. Conclusion (1 minute)

- **Speakers:** @yucpeng1
- **Content:**
  - Summarize the key points discussed.
  - Emphasize the contribution to HRI research and expected outcomes.
  - Thank the audience and invite questions.

Q&A Session (2 minutes)