Internship Projects



ZENDALONA

VISION BEYOND SIGHT

Date: 6-Nov-26
Version:1.0

Contents

Executive Summary 1

Internship Projects 2

I. z.Maths-Tutor V3 2

II. Liblouis-Table-Editor V3 4

III. z.PlayGarden - PC + MacOS 6

IV. Chess for Android (backend Lichess) 8

V. Rich Text Editor - Android 9

VI. Offline AI assistant for AC 26.04 11

VII. Music-Teacher Android/Windows/Mac OS 13

VIII. Telegram Accessibility V2 15

IX. AI Chatbot for Zendalona support 17

X. LMMS Accessibility 19

XI. System Wide Braille Input for Windows 21

Executive Summary

This document presents 11 open-source projects suitable for interns and students, with the scope and estimated duration of each. Undergraduate and postgraduate students with an interest in programming can participate. Teams will include varied roles such as business analysis, documentation, testing and marketing, and can draw on diverse backgrounds such as Computer Science, Humanities, BA and BSc Mathematics. There are no strict prerequisites: students should bring an open mind and a willingness to learn, with mentors guiding them on a regular, ideally daily, basis.

Every project produces a tangible result that helps visually impaired people around the world, especially in India through language-specific localization. If this phase succeeds, some projects may continue into follow-on work that adds enhancements and features to serve visually impaired users better.

Internship Projects

I. z.Maths-Tutor V3

z.Maths Tutor V3 aims to improve inclusive mathematics education by providing visually impaired learners with an accessible platform to develop mathematical skills. The enhancement will address limitations in question complexity, accessibility, language compatibility and user experience while expanding the range of practical mathematical concepts covered.

Requirements and Proposed Features

- Introduce structured questions with progressive difficulty levels, from basic to advanced.
- Add calculation dimensions covering time, currency, distance, direction and angles.
- Improve accessibility of menus, dropdowns and other interface components.
- Implement appropriate language mapping for interface elements and speech output.
- Fix existing functional bugs, including Indian language compatibility issues.
- Improve keyboard navigation and text-to-speech interaction.
- Provide an intuitive and consistent user experience.

Existing Features (Version 2)

- Desktop-based mathematics learning application primarily designed for visually impaired users.
- Keyboard-first navigation with integrated text-to-speech feedback.
- Structured learning pathways and quickplay sessions.
- Gamified challenges and auditory counting modes.
- Arithmetic operations including addition, subtraction, multiplication and division, along with percentage calculations.
- Multiple difficulty levels, configurable speech synthesizers and language settings.
- Support for English, Hindi, Malayalam, Tamil, Arabic and Sanskrit.

Recommended Technology

Python 3, GTK 3 (PyGObject), Pygame, GNU Gettext and speech synthesis integration (eSpeak NG).

Target Platforms

GNU/Linux, macOS and Windows.

Estimated Duration

3–6 months (development team of 3–5 members).

II. Liblouis-Table-Editor V3

Liblouis Table Editor V3 aims to simplify the creation, editing and maintenance of Braille translation tables. Liblouis is an open-source Braille translator and back-translator used by assistive technologies worldwide. Enhancing the editor will improve the reliability of table development, reduce configuration errors and make the process more efficient for developers and Braille language experts.

Requirements and Proposed Features

- Add provisions to specify nofor and noback opcodes directly within the editor.
- Implement quick documentation reference
- Fix the issue preventing newly added entries from being reflected in the main editor window after selecting an opcode and Unicode character.
- Introduce validation and error-prevention mechanisms to identify invalid table entries and configurations.
- Improve the reliability of adding, editing and managing translation table entries.

Existing Features (Version 2)

- Graphical interface for creating and editing Liblouis Braille translation tables.
- Support for opening existing .cti, .ctb and .utb files.
- Support for creating and managing translation rules.
- Real-time testing of translation rules.
- Exporting translation tables for use with Liblouis.
- File association and drag-and-drop support for opening table files.

Recommended Technology

Python 3, PyQt5, Liblouis.

Target Platforms

GNU/Linux, macOS and Windows.

Estimated Duration

3–6 months (development team of 2–3 members).

III. z.PlayGarden - PC + MacOS

Kindergarten level Games. The project aims to introduce computers as an accessible learning tool at an early age and make the learning experience engaging through gamification.

The application will focus on 3D audio, voice instructions and child-friendly games to help children understand basic concepts and learn about their surroundings through interactive activities. By combining spatial audio and guided learning, z.PlayGarden will provide an engaging environment for early learning and exploration.

Requirements and Proposed Features

- Develop a child-friendly learning application specifically for visually impaired Kindergarten children.
- Introduce basic computer interaction through engaging, game-based learning activities.
- Implement 3D audio to teach spatial concepts, including direction and distance.
- Provide voice instructions to guide children through the games and learning activities.
- Teach basic concepts such as left, right, height and weight.
- Help children identify animals and birds through their sounds.
- Introduce basic hygiene practices, including washing hands and maintaining cleanliness in their surroundings.
- Teach everyday words and object names, such as chair and blackboard, through interactive activities.

Target Platforms

PC (GNU/Linux and Windows) and macOS.

Estimated Duration

6–12 months (development team of 4–6 members).

IV. Chess for Android (backend Lichess)

Lichess is a widely used open-source chess platform. While its web version provides reasonable accessibility for visually impaired users, a similar experience is not readily available on mobile devices. This project aims to bridge this accessibility gap by integrating Lichess with a newly designed, fully accessible Android interface, enabling visually impaired users to play chess independently on mobile devices.

Requirements and Proposed Features

- Integrate the existing open-source Lichess platform to provide chess-playing functionality.
- Develop a new user interface specifically designed for visually impaired chess players.
- Implement a fully accessible frontend for mobile interaction.
- Provide comprehensive screen-reader support for interface elements and chessboard interaction.
- Implement voice feedback to communicate chessboard positions, pieces and game interactions.
- Ensure accessible selection, identification and movement of chess pieces.

Existing Accessibilty

The current web version offers reasonable accessibility support. This needs to be evaluated in detail and gaps needs to be bridged.

Target Platforms

Android.

Estimated Duration

3–6 months (development team of 1–3 members).

V. Rich Text Editor - Android

Visually impaired people currently lack an accessible Rich Text Editor on Android. This has been a long-pending request from the visually impaired community. Not everyone has regular access to a laptop, particularly students attending classrooms and people who need to work while travelling. Currently, users often rely on basic text editors to create documents on their Android devices. However, these editors do not provide the rich text formatting capabilities needed for further document editing. As a result, users have to transfer their documents to desktop applications to apply formatting and make additional changes. This creates an unnecessary dependency on desktop computers and limits the ability to create and edit properly formatted documents on the go.

The project aims to address this gap by developing an accessible Rich Text Editor for Android, enabling visually impaired users to create and format documents directly on their mobile devices.

Requirements and Proposed Features

- Develop a Rich Text Editor with accessibility as its primary focus.
- Ensure the editor and its controls are fully accessible to visually impaired users.
- Provide screen-reader-compatible access to editing and formatting operations.
- Support accessible navigation and interaction with the editor using assistive technologies.
- Provide basic text formatting operations, including bold, italic, underline and strikethrough.
- Support basic paragraph formatting, including headings, text alignment and lists.
- Ensure formatting operations can be performed and their effects understood through accessible feedback.

Target Platforms

Android.

Estimated Duration

3–6 months (development team of 3–5 members).

VI. Offline AI assistant for AC 26.04

Accessible Coconut (AC) is a GNU/Linux OS developed to provide an accessible computing environment for visually impaired people. It brings together accessibility technologies and applications to support independent computer usage.

This project aims to develop an AI assistant from the ground up, specifically designed to support Accessible Coconut 26.04. The assistant will work like an intelligent agent, allowing users to interact with the computer and perform actions through natural voice commands. For example, users should be able to dictate commands such as opening an application or starting to compose an email, rather than navigating through multiple interface elements manually.

A key requirement is that the assistant must operate entirely offline. This will allow visually impaired users, including those in remote locations, to access AI-assisted computer interaction without depending on an internet connection.

Requirements and Proposed Features

- Develop a new AI assistant from the ground up with support for Accessible Coconut 26.04.
- Provide an agent-like interaction model that can understand user instructions and perform the requested computer actions.
- Enable voice-based interaction so users can dictate actions using natural language.
- Support actions such as opening applications and starting tasks like composing an email through voice commands.
- Ensure the assistant operates offline without requiring an internet connection.
- Prioritise accessible interaction for visually impaired users.

Target Platform

Accessible Coconut 26.04 (GNU/Linux).

Recommended Technology

To be determined based on feasibility. This will need offline LLMs.

Estimated Duration

6–12 months (development team of 4–6 members).

VII. Music-Teacher Android/Windows/Mac OS

Most music-learning applications rely heavily on visual notation and interfaces, creating accessibility barriers for visually impaired learners. This project aims to address this gap by developing an accessible, self-instructional music tutor focused on teaching Carnatic music fundamentals. The application will use an audio-first and keyboard-driven approach, allowing learners to study musical concepts and practise independently without relying on visual notation.

The application will provide a structured learning experience covering svarams, musical notation and ragas. Through audio feedback, guided lessons, automatic self-correction and game-based practice, learners can engage with music lessons and improve their understanding through independent practice.

Requirements and Proposed Features

- Develop a cross-platform music-learning application using Python and Qt (PyQt/PySide).
- Ensure full keyboard accessibility and compatibility with screen readers.
- Provide audio feedback to announce svarams and musical notes.
- Implement structured lessons for learning Carnatic music fundamentals, including svarams, notation and ragas.
- Include automatic self-correction to help learners identify and correct mistakes during lessons.
- Provide a game-based practice mode to make learning more engaging.
- Prioritise audio-first interaction and accessible navigation throughout the application.

Recommended Technology

Python, Qt (PyQt / PySide).

Target Platforms

Windows, GNU/Linux and macOS.

Estimated Duration

3–6 months (development team of 2–3 members).

VIII. Telegram Accessibility V2

Zendalona has already released an accessibility-enhanced version of Telegram Desktop to improve its usability for visually impaired users. While the existing version provides desktop accessibility support, additional improvements are required to address specific user needs and make more Telegram features accessible.

This project aims to enhance the existing Telegram Desktop accessibility implementation by introducing additional capabilities and resolving identified accessibility issues. The focus will be on enabling visually impaired users to independently access voice recording, manage downloaded files, participate in group calls and interact with interface controls with appropriate screen-reader feedback.

Requirements and Proposed Features

- Enable visually impaired users to start and use voice recording independently.
- Improve accessibility of downloaded-file updates, allowing users to access relevant information about downloaded files.
- Implement full accessibility support for Telegram group calls, enabling visually impaired users to participate in them.
- Fix accessibility issues where certain buttons are not announced properly by screen readers.
- Improve the overall accessibility and usability of the existing Telegram Desktop implementation.

Existing Version

> Zendalona has already released an accessibility-enhanced version of Telegram Desktop.

Recommended Technology

C++, Qt, Telegram Desktop codebase.

Target Platforms

Desktop (Telegram Desktop).

Estimated Duration

6–12 months (development team of 4–6 members).

IX. AI Chatbot for Zendalona support

Zendalona develops products and assistive technologies to support visually impaired individuals. Users often need information, guidance and support to understand and use these products and technologies. An AI-powered support agent can make this information more accessible by providing conversational assistance and helping users find relevant information.

This project aims to develop and enhance an AI-powered agent dedicated to Zendalona product support and assistive technologies. It will use advanced Natural Language Processing (NLP) techniques to improve its ability to understand user queries and provide relevant responses. The project will also expand the agent's knowledge base to cover more information about Zendalona products and assistive technologies.

A key focus is multilingual support. Many users prefer communicating in their native languages rather than English. The chatbot will support Indic languages, including Hindi, Tamil, Telugu, Kannada and Bengali, as well as Spanish, French and Arabic. Users should be able to ask questions in their preferred language and receive responses in the same language.

Requirements and Proposed Features

- Develop an AI-powered agent for Zendalona product support and assistive technology information.
- Integrate advanced NLP techniques to improve query understanding and response quality.
- Expand the knowledge base with information about Zendalona products and assistive technologies.
- Improve accessibility features to ensure the chatbot is usable by visually impaired individuals.
- Implement multilingual interaction, allowing users to communicate in their preferred language.
- Support Hindi, Tamil, Telugu, Kannada, Bengali, Spanish, French and Arabic.
- Ensure responses are provided in the same language used by the user.

Recommended Technology

Python, NLP frameworks, Large Language Models (LLMs).

Target Platforms

To be determined based on the deployment approach.

Estimated Duration

6–12 months (development team of 4–6 members).

X. LMMS Accessibility

LMMS (Linux Multi Media Studio) is a digital audio workstation used for music production. However, it currently lacks adequate accessibility support for visually impaired users. This creates a barrier for people who want to learn music production, compose music and work creatively with digital audio tools.

Making LMMS accessible would provide visually impaired individuals with an opportunity to use a full music-production application and explore music creation independently. This project aims to improve the accessibility of LMMS so that visually impaired users can interact with its interface and access its music-production capabilities.

Requirements and Proposed Features

- Assess the existing LMMS application to identify accessibility barriers faced by visually impaired users.
- Improve screen-reader compatibility so that interface elements and controls can be identified and understood.
- Enable accessible navigation and interaction with the application's interface.
- Improve accessibility of the controls and features used for music production.
- Make the music-production workflow more accessible to visually impaired users.

Recommended Technology

C++, Qt, LMMS codebase.

Target Platforms

GNU/Linux.

Estimated Duration

3–6 months (development team of 2–3 members).

XI. System Wide Braille Input for Windows

Zendalona has developed iBusBraille, which provides a unified user experience for visually impaired users to interact with applications in the Linux operating system. It enables users to enter text using a six-key Braille typing method, supporting text input in different languages. However, this solution is currently available for Linux. Extending the same functionality to Windows as an open-source application will make six-key Braille input accessible to Windows users as well. This will allow visually impaired individuals to use a consistent Braille-based text input method while working across applications in the Windows environment.

Requirements and Proposed Features

- Implement iBusBraille for the Windows operating system, based on the existing Linux implementation.
- Develop the Windows version as an open-source application.
- Support six-key Braille typing for text input.
- Enable Braille-based text entry across applications in the Windows environment.
- Support text input in multiple languages.
- Maintain a consistent user experience with the existing Linux version.

Existing Features

- iBusBraille is already developed by Zendalona for Linux.
- Provides six-key Braille typing for text input.
- Supports text input across applications and in different languages.

Recommended Technology

To be determined based on the existing iBusBraille implementation and Windows integration requirements.

Target Platforms

Windows.

Estimated Duration

3–6 months (development team of 2–3 members).