CS 266 - Software Engineering Laboratory

Software Requirement Specification

WireWorks

Team Number: 28

Members:

202351001, Abhinav Chhajed

202351056, Kartik Gupta

202351142, Suraj Singh

202352319, Kavit Chugh

202352338, Viwan Jain

1.Introduction

1.1 Purpose

WireWorks aims to simplify circuit design accessibility for education and industry. This offline software provides an interactive platform for educators and students to learn and experiment with electronics, fostering fun learning experiences. Simultaneously, WireWorks empowers individual professionals and teams to streamline their circuit design documentation, improving clarity and collaboration. By simplifying these key processes, WireWorks seeks to accelerate innovation and knowledge sharing within the electronics community.

1.2 Document Conventions

Terminology:

"WireWorks": Refers to the software application being specified in this document.

"SRS": Refers to this Software Requirements Specification document.

"Component": Refers to the individual electronic parts (e.g., resistors, capacitors, microcontrollers) that users can incorporate into their circuit designs within WireWorks.

"Diagram": Refers to the visual representation of the circuit design within WireWorks, including block diagrams, circuit schematics, and PCB layouts.

1.3 Intended Audience and Reading Suggestions

This SRS targets Developers, Users, and Testers. Developers should focus on Sections 3 (Interfaces) and 4 (Features) for implementation details. Users should prioritize Sections 1 (Introduction), 2 (Overview), and 4 (Features) to understand software capabilities and limitations. A suggested reading order is Sections 1-2 for all, then Section 4 for Users and, finally the entire document for Developers.

1.4 Product Scope

WireWorks provides a user-friendly, offline software environment for creating, verifying, and documenting circuit designs. The software's goal is to simplify the design process for students, educators, and hobbyists. Key objectives include an interactive GUI, wide component library (Arduino, ESP, Adafruit), automated connection correctness verification, and diagram conversion between different diagram types. Benefits are increased efficiency, accuracy, and accessibility. This project provides practical experience in software development and electronics. The scope will not include features like real-time collaboration, advanced simulation or external database for third-party component integration as it compromises software reliability.

1.5 References

1. IEEE 830-1998 Standard for Software Requirements Specifications

2.Overall Description

2.1 Product Perspective

WireWorks is a new, self-contained, cross-platform desktop application built with Python and PyQt5 which works on distributions like Windows, macOS, Linux. It is a standalone desktop application with an intuitive UI, enabling engineers, educators, students, and professionals to design circuit diagrams effectively. It integrates diagram conversion, component management, and export functionalities along with connection correction verification algorithms.

As an independent software product, WireWorks does not require external dependencies. While future integration with third-party tools is possible, it currently functions independently. The application streamlines the circuit design process, making it accessible for both beginners and experienced users.

A system architecture diagram illustrating major components and interfaces will be included in a later section.

2.2 Product Functions

WireWorks provides essential functionalities for circuit design, enabling users to create, edit, and manage electrical circuit diagrams efficiently. The major functions include:

Template Management – Create and manage reusable circuit templates.

Circuit Design – Design circuits with an interactive UI.

Component Selection & Arrangement – Choose from a library of components and place them in the workspace.

Component Connections – Connect components logically to form complete circuits.

Project Management – Create, rename, open, and save projects.

Diagram Conversion – Convert between schematic, block, and PCB layout views.

Export Functionality – Save diagrams in multiple formats such as PNG, PDF, and JPEG.

Component Library Updates – Keep component data up to date.

Customization – Adjust grid settings and themes for personalized experience.

Feedback & Updates – Allow users to push feedback and receive software updates.

2.3 User Characteristics

WireWorks is designed for a diverse range of users involved in circuit design:

Engineers – Professionals designing electrical circuits for various applications.

Students – Learners studying circuit design and electronics.

Industry Professionals – Users managing large-scale electronic projects.

Each user class requires different levels of functionality, with experienced engineers benefiting from advanced features like component library customization, while students can use pre-built templates for learning.

2.4 Operating Environment

WireWorks is designed to run efficiently across multiple platforms, ensuring accessibility and ease of use:

Supported Operating Systems: Windows, macOS, Linux.

System Requirements: Minimal RAM and processing power for smooth operation.

Offline Functionality: Can work without an internet connection,projects are stored locally**.**

2.5 Design and Implementation Constraints

The development of WireWorks is subject to the following constraints:

Cross-Platform Compatibility – Must function seamlessly on Windows, macOS, and Linux.

File Export Requirements – Must support PNG,PDF, JPEG for external usage.

Offline – Should operate offline.

Scalability & Performance – The application should handle large circuit designs efficiently which include a wide variety of electronic components and should be able to manage connection validation.

2.6 User Documentation

The following documentation will be provided with WireWorks:

User Manual – Step-by-step guide on using the software.

Online Help & FAQs – Integrated help section for troubleshooting.

Tutorials – Interactive guides for beginners.

2.7 Assumptions and Dependencies

Users are expected to have basic knowledge of circuit design principles as we do not intend to teach electronics to the user, we just provide them with a software tool to put their existing knowledge to use and easily create various diagram types to document their projects.

3. External Interface Requirements

3.1 User Interfaces

The Wire Works software will provide an interactive and user-friendly graphical interface for circuit design and project management.

User Interface Features:

Dashboard: Displays recent projects and quick access to tools.

Diagram Workspace: A drag-and-drop interface for designing circuits.

Menu: Provides options for adding components, saving projects, exporting diagrams, and managing settings.

Component Library: A categorized list of available circuit components.

Settings Panel: Users can configure grid settings, themes, export formats, and other preferences.

Zoom & Pan: Allows users to navigate and adjust their circuit diagrams efficiently.

3.2 Hardware Interfaces

Wire Works will be compatible with various hardware configurations to ensure seamless user experience.

Minimum Hardware Requirements:

Processor: Intel Core i3 or equivalent

RAM: 4GB (8GB recommended for large projects)

Storage: 500MB of free disk space for installation

Graphics: Integrated GPU with OpenGL support for rendering

Input Devices: Keyboard, mouse, or touchscreen for interaction

The software should support external drawing tablets for a better circuit sketching experience.

3.3 Software Interfaces

Wire Works will interact with other software systems to improve functionality.

Operating System Compatibility:

Windows (10 and above)

macOS (latest versions)

Linux distributions (Ubuntu, Debian, etc.)

File Formats Supported:

Export: PNG, JPEG, PDF.

3.4 Communications Interfaces

Wire Works will have limited communication requirements but may include the following:

Local Storage.

Projects will be stored locally by default.

Version Updates: The software will check for updates online.

4. System Features

4.1 Circuit Design Module

4.1.1 Description

Enables users to create and modify circuit diagrams with a drag-and-drop interface for easy component placement.

4.1.2 Stimulus/Response sequences

Stimulus: The user selects a component from the library and places it on the workspace.
Response: The component appears on the interface, ready for positioning and connections.

Stimulus: The user connects two components using a circuit line.
Response: A connection is established, visually represented by a wire.

Stimulus: The user presses undo or redo.
Response: The last action is either reverted or reapplied.

4.1.3 Functional Requirements

Add, remove, and label components.

Connect components using circuit lines.

Undo and redo actions.

4.2 Project Management

4.2.1 Description

Allows users to create, save, rename, and open projects for efficient organization and workflow management.

4.1.2 Stimulus/Response sequences

Stimulus: The user creates a new project.
Response: A new, blank workspace is initialized.

Stimulus: The user saves or renames a project.
Response: The project is saved with the given name and stored for future access.

Stimulus: The user closes the application without saving changes.
Response: A prompt appears asking whether to save before exiting.

4.1.3 Functional Requirements

Manage multiple projects simultaneously.

Auto-save feature to prevent data loss.

4.3 Diagram Export Module

4.3.1 Description

Enables users to export circuit diagrams in multiple formats for documentation and sharing.

4.3.2 Stimulus/Response sequences

Stimulus: The user selects an export option and chooses a file format.
Response: The system processes the diagram and exports it in the selected format.

4.3.3 Functional Requirements

Select preferred file format (PNG,PDF, JPEG).

Adjust resolution and quality settings before exporting.

4.4 Component Management

4.4.1 Description

Allows users to update and add new circuit components for

customization and reuse.

4.4.2 Stimulus/Response sequences

Stimulus: The user modifies a component’s properties.
Response: The system updates the component attributes accordingly.

4.4.3 Functional Requirements

Modify component properties as needed.

Store frequently used components in a library.

4.5 Diagram Conversion

4.5.1 Description

Facilitates conversion between schematic, block, and PCB layouts while maintaining circuit integrity.

4.5.2 Stimulus/Response sequences

Stimulus: The user selects an option to convert the diagram into another layout.
Response: The system reconfigures the diagram while maintaining

circuit integrity.

Stimulus: The user switches from schematic to PCB layout.
Response: The system adjusts the component positioning while preserving connections.

4.5.3 Functional Requirements

Automatically adjust component positioning during conversion.

Maintain circuit logic when switching between layouts.

4.6 User Interface Customization

4.6.1 Description

Provides options to modify UI settings for an optimized user experience.

4.6.2 Stimulus/Response sequences

Stimulus: The user changes the grid settings or theme.
Response: The workspace updates to reflect the new settings.

Stimulus: The user enables or disables a toolbar.
Response: The specified toolbar appears or disappears accordingly.

4.6.3 Functional Requirements

Change grid settings, themes, and UI layout.

Enable or disable specific toolbars and menus.

4.7 Feedback & Updates

4.7.1 Description

Allows users to submit feedback and receive software updates to enhance functionality and usability.

4.7.2 Stimulus/Response sequences

Stimulus: The user submits feedback through the built-in form.
Response: The system stores the feedback and notifies the development team.

Stimulus: The system detects a new update.
Response: A notification appears, prompting the user to install the update.

4.7.3 Functional Requirements

Collect feedback via a built-in form.

Notify users about new updates.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system must ensure high responsiveness by minimizing delays during key operations, such as error validation and diagram conversion. Real-time error detection should be achieved through a fast error validation algorithm, allowing users to quickly identify and correct mistakes. Additionally, the modular architecture should guarantee that performance remains optimal even under heavy system load, supporting scalability for future growth and enhancements.

5.2 Reliability Requirements

To maintain the stability and accuracy of circuit designs, the system must provide reliable connectivity validation mechanisms. This ensures secure and correct connections between components. Moreover, it should consistently produce accurate component metadata and diagram conversions, preserving the integrity of the system's outputs. Reliability must be sustained over time, ensuring continuous and predictable system behavior.

5.3 Usability Requirements

The system must offer a user-friendly interface with an intuitive design to enhance the user experience. Features such as a simple drag-and-drop mechanism and an organized component library will facilitate easy circuit creation. To further improve usability, distinguishable wires, high-resolution component images, and clear labels should be incorporated, making component connections and circuit visualization more straightforward and efficient for users.

5.4 Extensibility Requirements

Extensibility is a critical requirement for fostering system evolution and growth. The tool must support easy extension of the component library, enabling both developers and users to add new components seamlessly.

5.5 Software Quality Attributes

The system architecture must be highly modular to support decomposable and reusable components, ensuring efficient maintenance and enhancement. Maintainability is essential, with well-structured code that allows for easy bug fixes and updates. Scalability must also be a priority to accommodate future growth and increased user demands.

5.6 Business Rules

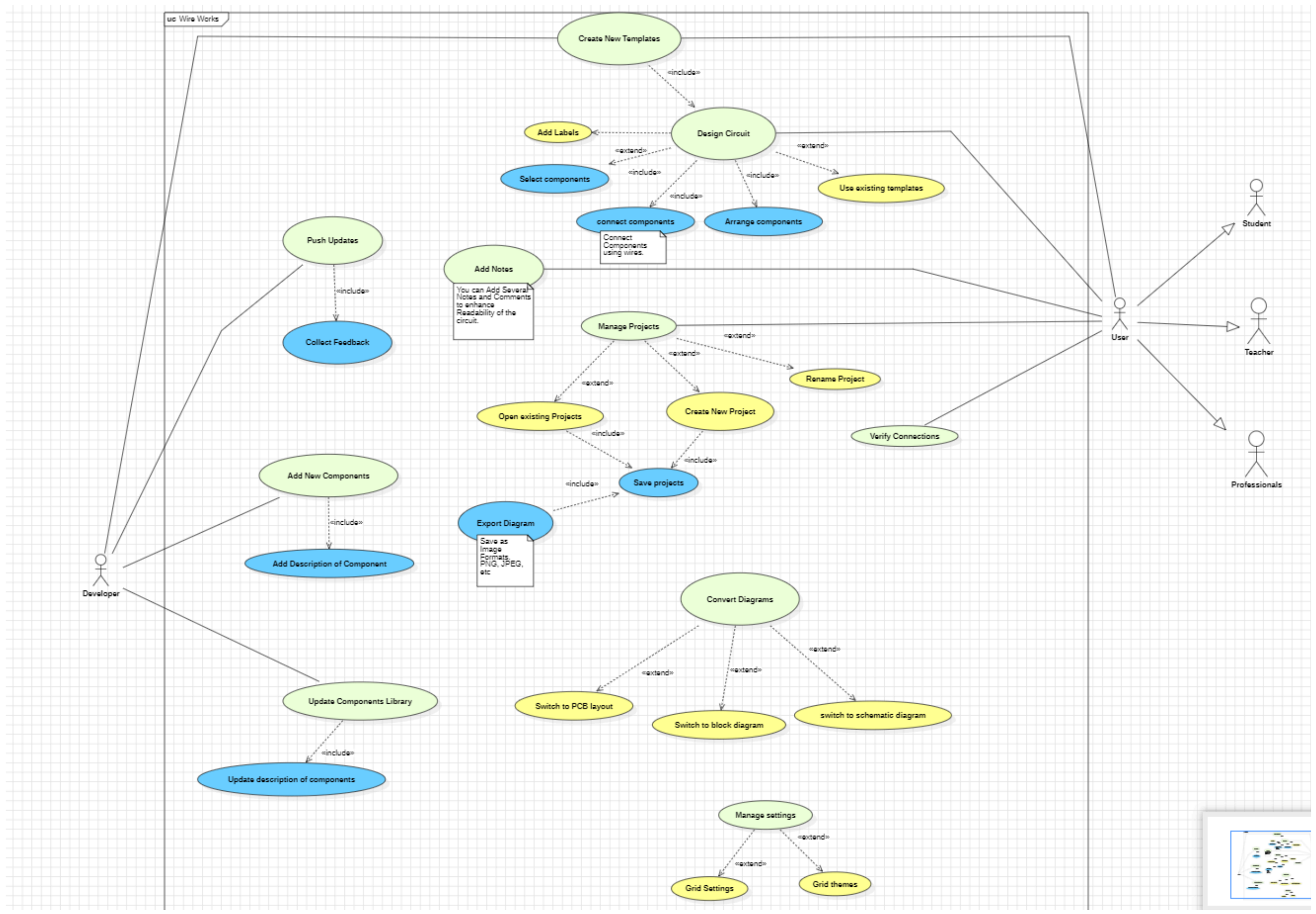
Developers must maintain a version-controlled component library to track changes and improvements. Additionally, the tool should adhere to industry-standard practices for electronic component design, ensuring compliance and user confidence in the system's outputs.

6. Other Requirements

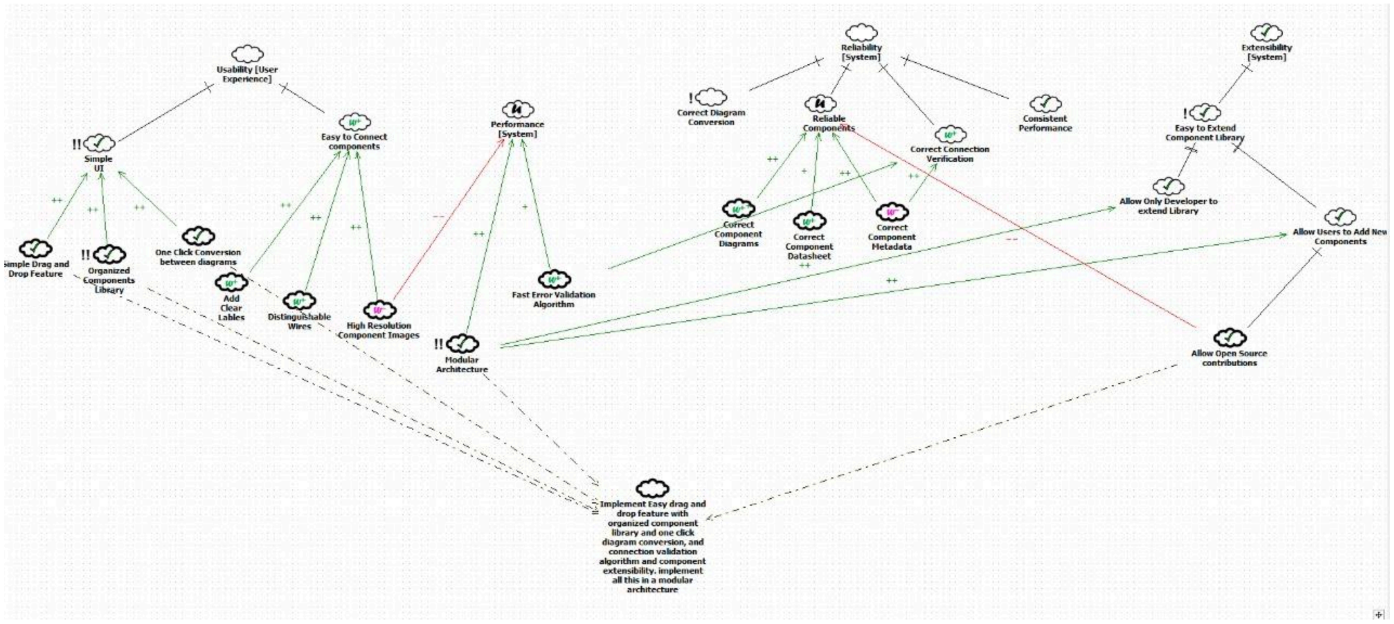
Appendix A: Glossary

//add whole in gpt and ask to make glossary

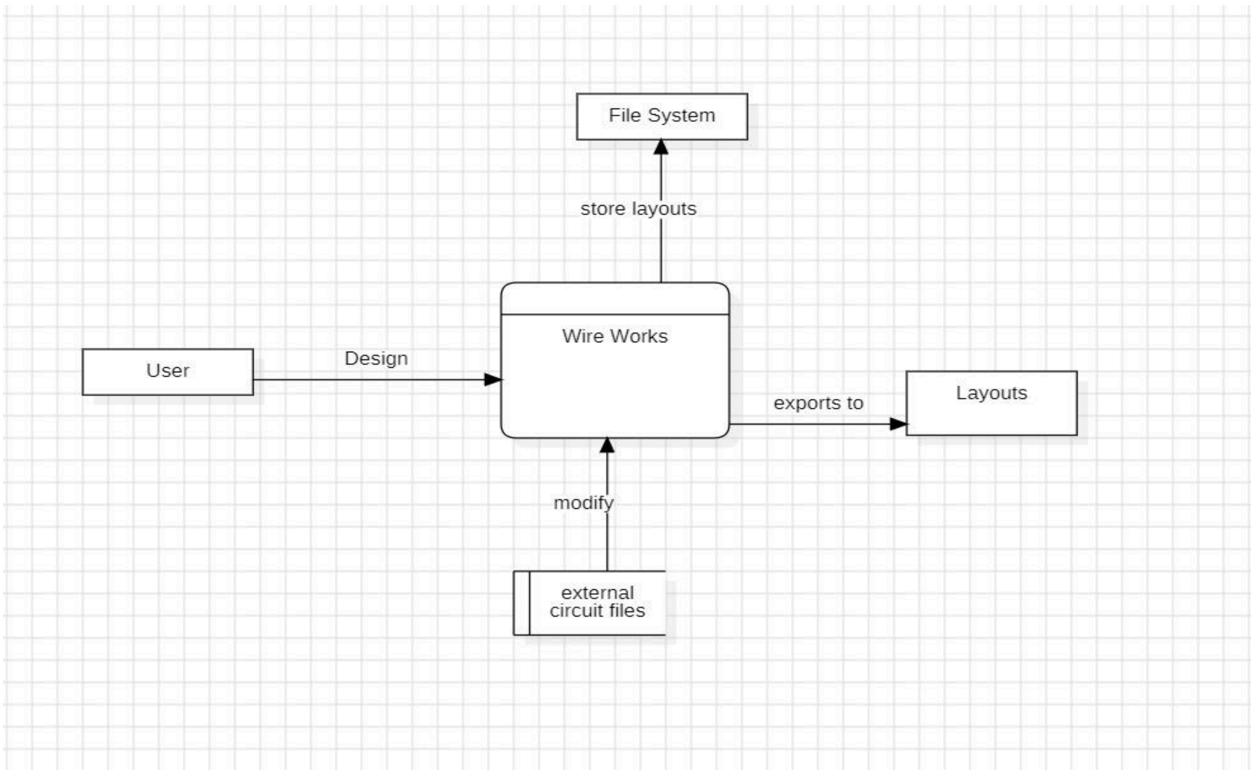
Appendix B: Analysis Models



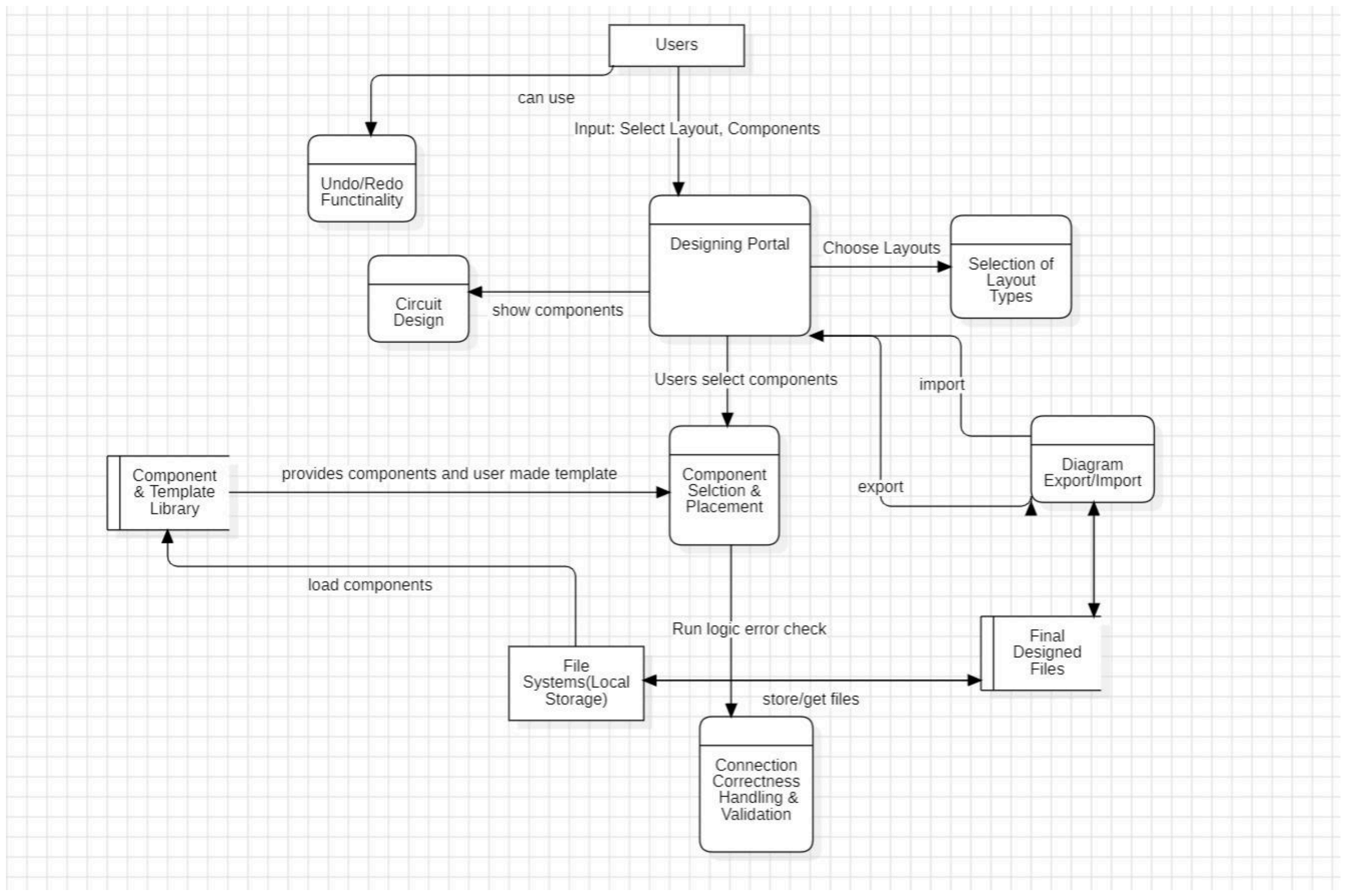
Strategic Rationale Diagram


Usability [User Experience]
Simple UI
Easy to Connect components
Performance [System]
Correct Diagram Conversion
Reliability [System]
Reliable Components
Correct Connection Verification
Consistent Performance
Extensibility [System]
Easy to Extend Component Library
Allow Only Developer to extend Library
Allow Users to Add New Components
Simple Drag and Drop Feature
Organized Components Library
One Click Conversion between diagrams
Add Clear Lables
Distinguishable Wires
High Resolution Component Images
Fast Error Validation Algorithm
Modular Architecture
Correct Component Diagrams
Correct Component Datasheet
Correct Component Metadata
Allow Open Source contributions
Implement Easy drag and drop feature with organized component library and one click diagram conversion, and connection validation algorithm and component extensibility. implement all this in a modular architecture

Non-functional- Soft Goal Interdependence Diagram



Data Flow Diagram - Level 0


Users
can use
Input: Select Layout, Components
Undo/Redo Functinality
Designing Portal
Choose Layouts
Selection of Layout Types
Circuit Design
show components
Users select components
import
Component & Template Library
provides components and user made template
Component Selction & Placement
export
Diagram Export/Import
load components
Run logic error check
File Systems(Local Storage)
store/get files
Final Designed Files
Connection Correctness Handling & Validation

Data Flow Diagram - Level 01

Appendix C: To Be Determined List

"There are no tbd items in this document."