CodEZ

CodEZ, Team 5

Stetson Baldwin, Owen Newberg, Malcolm Zartman, Alexander Stotts

Software Requirements Specification Document

Version: (2) **Date: (05/12/2025)**

Table of Contents

1 Purpose

Nearly half of all job postings (49%) require some form of coding skills, but unfortunately Computer Science has the highest dropout rate of any major at 9.8%. As a team of current or past tutors to introductory programming courses, we have seen many students getting caught up on the tools they use rather than the code itself. CSUSM has used Empress, a virtual desktop for school-related programs for over 25 years, it served the school well offering students an opportunity to learn both the UNIX operating system in the command line and the technical knowledge required to run programs manually. However, for first-semester freshmen, with little to no experience, learning how to connect to a virtual machine and run code in the UNIX OS can be complicated. Replit, an easy-to-use online IDE, was used by several professors to help freshmen run their code without learning the old system. However, in 2023 it was revealed that Replit was using student data to train an AI model, since it was unveiled, the university has returned to Empress. After the return, our team has seen firsthand how the Empress system has negatively impacted how students learn to code. Having to learn something new in a Unix-based system as well as coding itself, has been too overwhelming for many college aged students. In comes the need for a replacement to an application that is efficient, user-friendly, secure, and most importantly it needs to make learning code easy. Not to mention, this application can also be embedded into websites or other applications, allowing for any organization to run code from any machine anywhere in the world so long as they have internet access.

2 Scope

Our application will be named CodEZ because we want to make CodEZ, a platform that any student, no matter how little technical knowledge they possess, can use our platform and pick up coding easy for any language. Specifically, we want to add these features to CodEZ: Developing a Minimalist, User-Friendly Interface Goal: The focus will be on the basic coding functions like editor, file management and console output with advanced features made accessible to the user only when he or she is ready for them. Outcome: This will decrease the initial learning curve so that new programmers can start coding without being overwhelmed. Creating a Seamless Setup Experience Goal: To provide a one click (or few click) installation process that hides the complexity of system configurations and library dependencies. Outcome: This enables beginners to get started in minutes, thus lowering the threshold for entry. Provide Clear Error Reporting and Debugging Goal: To develop a simple debugging panel that shows errors, states their possible causes and suggests solutions in easy-to-understand terms. Outcome: To help new programmers learn how to troubleshoot on their own and gain confidence in their ability to solve problems. Ensure Lightweight Performance Goal: To optimize the IDE's core to run well on the typical student's laptops and budget hardware. Outcome: To offer stable performance which will keep users coming back even on low-end devices. Plan for Expandability Goal: To enable developers to add optional features like version control and advanced refactoring over time as the developers gain experience. Outcome: To enable a path from beginner to intermediate to advanced developer within the same environment and without having to switch between different tools. And we will design this application while keeping the privacy and security of the user at the highest priority.

3 User characteristics

3.1 Key users

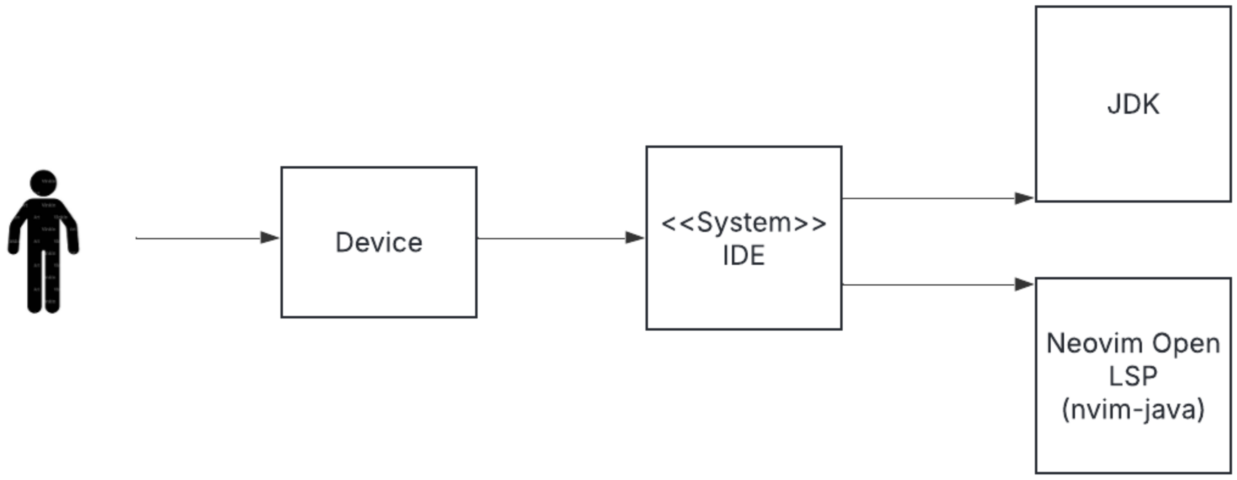
Key users of CodEZ are novice college students who are just starting to learn coding. Most have little to no experience with programming and are unfamiliar with command-line tools or system configurations. They often find installation processes frustrating and prefer a straightforward, ready-to-use coding environment. These users primarily range from high school to college students but can include individuals of all ages who are new to coding or need a quick refresher. Since they are still developing their technical skills, the platform must be user-friendly, minimizing setup difficulties and allowing them to focus on writing and understanding code.

3.2 Secondary users

Secondary users are journeyman and masters of coding, which include professors, teaching assistants, tutors, family members, and supervisors. These individuals already have knowledge of coding and typically serve in a guiding or evaluative role rather than as primary learners. Their interactions with the platform involve assisting key users, reviewing code, or troubleshooting issues. While they may appreciate a smooth and efficient experience, their needs take a backseat to those of key users when designing features and interface elements.

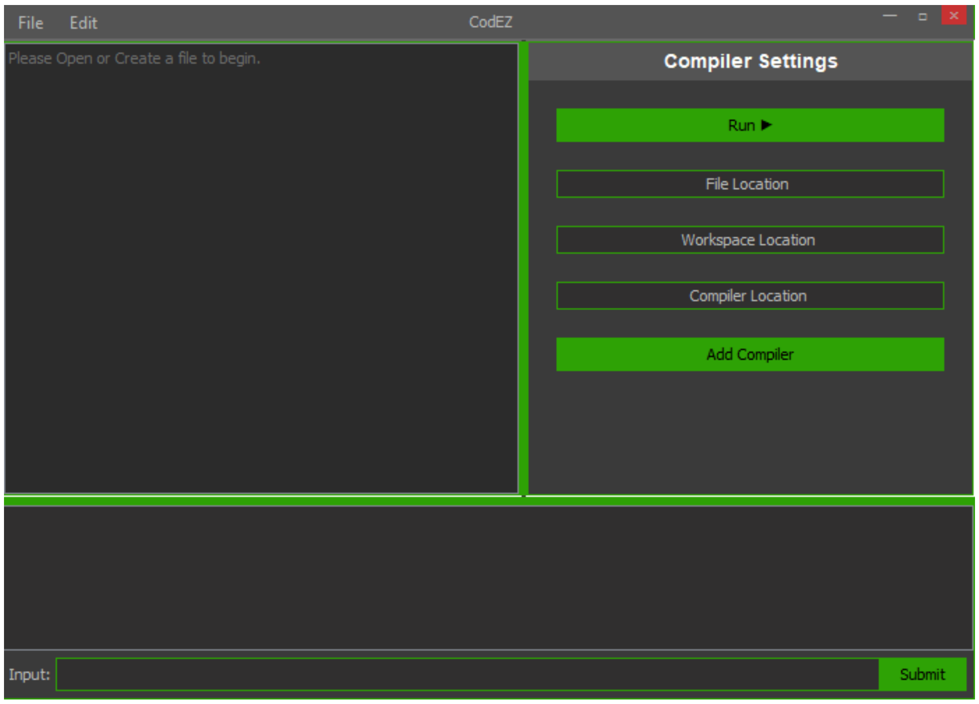
4 Product perspective

4.1 System Context



4.2 User interfaces

- It is required that the system provides a Terminal Output and Input Textbox.
- It is required that the system provides clickable TextLabels to show files and access them.
- It is required that the system provides an Input textbox for writing code.
- It is required that the system provides a run button which will compile and run code at a file location specified by the user through typing in the file location and workspace location text field or through opening a file.
- It is required that the system provides a save button to update and save the file.
- It is required that the system provides text highlighting.
- It is required that the system provides a new button to create a new file and save it to a certain destination.
- It is required that the system interfaces have the following screen layout:



4.3 Software interfaces

- The system will use JRE for development
- The system will use the JDK compiler to compile Java code.
- The system will use Neovim Open LSP (nvim-java) for text highlighting of Java files.

- The system will use the Windows operating system to assist in creating, saving, and opening Java files.

4.4 Deployment requirements

- The user must download the application.
- The user must have a Windows OS installed.
- A CPU with over 1.4 GHZ of processing speed, 2 cores, 8GB of ram and at least 1GB of empty space on their hard drive.
- It is recommended that the user have a 2.4 GHZ 4 core CPU and 16 GB of ram if they hope to compile and run their code optimally.
- The user must have the JDK-11 compiler, for java, installed.

5 Assumptions and Dependencies

- We assume that user has amateur experience with coding
- We assume that user is using a Windows system
- We assume that the user has basic knowledge on computers(locating files and navigating their system).
- We assume user plans on using java.
- We assume that user has access to internet to download compiler binaries

6 Specific requirements

6.1 System Functional Requirements

- As a programmer I need to be able to save files, so I can always have access to my code.
- As a programmer I need to be able to open my files in the editor, in order to access my files.
- As a programmer I need text highlighting, so I can more easily understand the structure of my code and identify mistakes.
- As a novice programmer I want a run button, so I don't have to memorize compile or run commands of various programming languages.
- As a programmer I want to adjust the panels of the IDE's UI, so I can have more personalization with the IDE.
- As a programmer I need error detection, so I can fix errors more easily.

6.2 Logical Database Requirements

- N/A

6.3 Software System Attributes

- As a student, I want each component in the IDE UI to be resized such that it can fit into one page, so that I can easily view my code and files.
- As a programmer user I need a lightweight (performant) and rapid access to my code.
- As a professor I want to allow my students to run Java code without students being significantly more than intermediately skilled with the Windows system.
- As a CSUSM student Empress offers Vim and another IDE however there is no mouse support and the list of commands to control and edit in these IDEs is too extensive for me to learn.
- As a CSUSM student Empress quickly becomes complicated and the putty connection disconnects frequently, I need to program without disconnecting from my IDE or crashing.
- As a Professor at CSUSM I need my students to be able to run Code for Java with as little hassle as possible.

6.3.1 Usability

- Students will be able to use and understand CodEZ after the first week of class.
- Teachers will be able to use and understand CodEZ in half an hour.
- Novice programmers will be able to compile and run files without having any knowledge of terminal commands.
- Students shall be able to save Java files onto their Windows device with zero issues after learning to use CodEZ.

6.3.2 Performance

- The system shall be compiled and run in O(n) 100% of the time, unless an exception is created by the user.
- The system shall be kept under a Gigabyte to prevent long download or open load times.
- The system shall never search in O(n^2) when searching for values
- The system shall only execute in O(n^2) when explicitly required or (when the operational cost is below 30ns per element ‘n’ and total elements ‘n’ < 100.

6.3.3 Reliability/Dependability

- This program will be able to run on Windows.
- The system shall fail gracefully, if at all.
- The program will not run the code if it detects and deems that the code attempting to be run is malicious.
- The code will not be saved if the device crashes the program and/or the device's battery becomes fully depleted.

6.3.4 Security

- The system will not require any authentication due as there is no intentions of collecting personal information
- The system will save logs to a separate file which is deleted when the program is successfully terminated only saving those logs when the program has crashed or reached an exception
- The system will obfuscate the code before publishing to prevent source code from being repackaged maliciously and used to break into users' systems.
- The users' data will only be available to the user themselves on their local machine in all other cases where data might be exposed to the internet, data will not be stored.
- The system will research potential security risks related to the java runtime and do our best to prevent any such cases from being exposed in our program such as buffer overflow attacks, and other malicious injections

6.3.5 Maintainability

- N/A