# Number System Converter (PWA) — Documentation

## ii. Process Documentation

Project Title: Automation of Number System Conversions from One Number System to Another

### Objective:

To build a web-based Progressive Web App (PWA) that automates the conversion of numbers between Binary, Octal, Decimal, and Hexadecimal systems.

### Team :

Group of five members;

1.ArnoldArnold Toby Thuranira

CT101/G/22183/24

2.Cosmas Ongode

Ct101/g/22105/24

3.Anne Kerubo

CT101/G/22698/24

4.Anyasi Liloyo Munala

CT101/G/22812/24

5.Nkonge William Mutethia

Ct101/G/22542/24

6.Jele Dayib Idris

CT101/G/22738/24

### Tools and Technologies:

HTML5 – for webpage structure  
CSS3 – for styling and layout  
JavaScript (ES6) – for logic and interactivity  
PWA (Progressive Web App) – to enable installation on Android/iOS  
GitHub – for version control and collaboration  
VS Code – as code editor  
Chrome DevTools – for testing

### Process Steps:

* Planning Phase: Defined objectives, formed group, and assigned roles.
* Design Phase: Created responsive UI and designed conversion interface.
* Development Phase: Implemented logic using JavaScript and regex validation.
* Testing Phase: Tested various conversions and ensured responsive design.
* Deployment Phase: Hosted via GitHub Pages and tested on Android/iOS.
* Collaboration: Shared repository and added lecturer as collaborator.

### Expected Output:

A fully functional web app capable of converting between number systems, installable as a PWA on mobile devices.

## iii. Program Documentation

### Project Name:

Number System Converter (PWA)

### Description:

This program allows users to input a number in one base (Binary, Octal, Decimal, or Hexadecimal) and instantly convert it into all other bases. It’s developed as a Progressive Web App (PWA) for offline and mobile use.

### Functional Requirements:

* Accept user input for a number.
* Allow selection of 'from' and 'to' number systems.
* Perform accurate conversions.
* Display equivalent values for all systems.
* Handle invalid inputs gracefully.
* Be installable on mobile devices.

### Non-Functional Requirements:

* Responsiveness: Works on phones, tablets, and desktops.
* Offline Availability: Works offline using service worker.
* Usability: Simple UI and layout.
* Performance: Fast conversion using BigInt.

### System Design:

Input: User enters a number (e.g., 1010) and selects a base (e.g., Binary).

Processing: Validate input, convert to decimal, then reconvert to all bases.

Output: Display converted values for Binary, Octal, Decimal, and Hexadecimal.

### Program Flow (Simplified):

User Input → Validation → Decimal Conversion → Base Conversions → Display Results

### Key Functions:

function parseInput(value, base) { ... } // Validates and parses input  
function toBaseString(n, base) { ... } // Converts number to target base  
function convertAll(input, fromBase) { ... } // Converts to all bases  
function doConvert() { ... } // Main handler on button click

### Error Handling:

Alerts user for empty or invalid input. Uses try...catch to handle runtime errors.

### Deployment:

Hosted via GitHub Pages, registered as a PWA with service-worker.js, installable on Android and iOS.