---

3. TypeScript core library files

3.1 `src/protocol/wavelengthMap.ts`

```ts
// src/protocol/wavelengthMap.ts

/**
 * Mapping between alphabet letters, their hex color representation,
 * and a canonical wavelength (in nanometers) across the visible spectrum.
 */

export type LetterSymbol = {
  letter: string;
  hexColor: string;
  wavelengthNm: number;
};

const LETTER_COLORS: string[] = [
  "#8B00FF", // A - deep violet
  "#7A00FF", // B
  "#6900FF", // C
  "#5800FF", // D
  "#4700FF", // E
  "#3600FF", // F
  "#2500FF", // G
  "#1400FF", // H
  "#0300FF", // I
  "#0040FF", // J - blue
  "#0055FF", // K
  "#006AFF", // L
  "#0080FF", // M - cyan-ish
  "#00A0FF", // N
  "#00C0FF", // O
  "#00E0FF", // P
  "#00FFB0", // Q
  "#40FF40", // R - green
  "#80FF00", // S
  "#AFFF00", // T
  "#DFFF00", // U
  "#FFFF00", // V - yellow
  "#FFBF00", // W
  "#FF8000", // X - orange
  "#FF4000", // Y
  "#FF0000", // Z - red
];

/**
 * Evenly distribute wavelengths across the visible spectrum
(approx 380–740 nm)
 * for the 26 letters A–Z.
 */
const LETTER_WAVELENGTHS: number[] = (() => {
  const min = 380;
  const max = 740;
  const count = 26;
  const step = (max - min) / (count - 1); // 360 / 25 = 14.4
  const arr: number[] = [];
  for (let i = 0; i < count; i++) {
    arr.push(Math.round(min + step * i));
  }
  return arr;
})();

/**
 * The full alphabet map A–Z.
 */
export const ALPHABET_MAP: LetterSymbol[] = (() => {
  const symbols: LetterSymbol[] = [];
  for (let i = 0; i < 26; i++) {
    const letter = String.fromCharCode("A".charCodeAt(0) + i);
    symbols.push({
      letter,
      hexColor: LETTER_COLORS[i],
      wavelengthNm: LETTER_WAVELENGTHS[i],
    });
  }
  return symbols;
})();

/**
 * Lookup table from letter to symbol.
 */
const LETTER_TO_SYMBOL = new Map<string,
LetterSymbol>(
  ALPHABET_MAP.map((s) => [s.letter, s])
```

```
);

/**
 * Get the symbol info (letter, hexColor, wavelengthNm) for a
given letter.
 */
export function getLetterInfo(letter: string): LetterSymbol |
undefined {
  if (!letter) return undefined;
  const upper = letter.toUpperCase();
  return LETTER_TO_SYMBOL.get(upper);
}

/**
 * Get the canonical wavelength (nm) for a given letter.
 */
export function getWavelengthForLetter(letter: string):
number | undefined {
  const info = getLetterInfo(letter);
  return info?.wavelengthNm;
}

/**
 * Find the nearest defined letter for a given wavelength.
 * Returns null if alphabet is empty.
 */
export function getLetterForWavelength(
  wavelengthNm: number
): string | null {
  if (ALPHABET_MAP.length === 0) return null;
  let best: LetterSymbol = ALPHABET_MAP[0];
  let bestDiff = Math.abs(ALPHABET_MAP[0].wavelengthNm -
wavelengthNm);

  for (let i = 1; i < ALPHABET_MAP.length; i++) {
    const candidate = ALPHABET_MAP[i];
    const diff = Math.abs(candidate.wavelengthNm -
wavelengthNm);
    if (diff < bestDiff) {
      best = candidate;
      bestDiff = diff;
    }
  }

  return best.letter;
}
```