**Functional Requirements – Solar Module Visualization App**

1. **Geometry Input ✅** 
   * The app **should receive** the following inputs from the user:
     + Module width and height
     + Front and rear glass thickness
     + Front and rear encapsulant thickness
     + Frame thickness
     + Number of cell rows and columns
     + Cell spacing
     + Cell dimensions (from a dropdown list)
2. **View Management ✅**
   * The app **should provide** a dropdown menu to select different viewing perspectives.
   * The available views **should include**: Top, Side, Front, Bottom, and Isometric.
   * Selecting a view **should update** the 3D visualization accordingly using orthographic or perspective projection as appropriate.
3. **3D Visualization ✅**
   * The app **should draw** a 3D model of the solar module composed of:
     + Rear glass
     + Rear encapsulant
     + Photovoltaic cells
     + Front encapsulant
     + Front glass
   * Each layer **should be stacked** in the correct Z-order based on user-defined thicknesses.
   * Cells **should be placed** in a grid based on the number of rows and columns, centered and evenly spaced.
   * In Top view, cells **should be drawn** using 2D rectangle() objects to avoid visual artifacts.
4. **Cell Layout Constraints ✅**
   * If the specified number of cells cannot fit within the module dimensions, the app **should display** a uialert warning showing the maximum rows and columns that can be drawn.
5. **Frame ✅**
   * The app **should include** an E-shaped aluminum frame modeled using multiple stacked L-strips.
   * A checkbox **should control** frame visibility.
   * When the checkbox is unchecked, the frame **should not be rendered**.
6. **Irradiance Input and Cell Color ✅**
   * The app **should receive** a manual irradiance value (in W/m²) via a numeric edit field.
   * The color of the cells **should change** dynamically based on irradiance using a color gradient from blue (low irradiance) to red (high irradiance).
   * A color legend image **should be included** to illustrate the mapping from irradiance to color.
7. **Power Calculation ✅**
   * The app **should calculate** power using the corresponding formula:
   * The app **should receive** efficiency as a user input (default 0.18).
   * The computed power value **should be displayed** in the plot area or as a figure title.
8. **Configuration Save/Load ✅**
   * The app **should allow** the user to save all current parameters to a .mat file.
   * The app **should be able to load** a previously saved configuration and apply it to the interface and visualization.
   * Saved parameters **should include** geometry, cell layout, efficiency, irradiance, frame visibility, and camera view.