# > OBJETIVOS DE APRENDIZAJES

## CARD VALIDATION

**HTML y CSS**

- ☐ Uso de HTML semántico.
- ☐ Uso de selectores de CSS.
- ☐ Construir tu aplicación respetando el diseño realizado (maquetación).

**DOM**

- ☐ Uso de selectores del DOM.
- ☐ Manejo de eventos del DOM.
- ☐ Manipulación dinámica del DOM. (appendChild |createElement | createTextNode| innerHTML | textContent | etc.)

**JavaScript**

- ☐ Manipulación de strings.
- ☐ Uso de condicionales (if-else | switch | operador ternario)
- ☐ Uso de bucles (for | for..in | for..of | while)
- ☐ Uso de funciones (parámetros | argumentos | valor de retorno)
- ☐ Declaración correcta de variables (const & let)

**Testing**

- ☐ Testeo unitario.

**Estructura del código y guía de estilo**

- ☐ Organizar y dividir el código en módulos (Modularización)
- ☐ Uso de identificadores descriptivos (Nomenclatura | Semántica)
- ☐ Uso de linter (ESLINT)

**Git y GitHub**

- ☐ Uso de comandos de git (add | commit | pull | status |push)
- ☐ Manejo de repositorios de GitHub (clone | fork | gh-pages)

**UX**

- ☐ Diseñar la aplicación pensando y entendiendo al usuario.
- ☐ Crear prototipos para obtener feedback e iterar.
- ☐ Aplicar los principios de diseño visual (contraste, alineación, jerarquía)

## CIPHER

**HTML y CSS**

- ☐ Uso de HTML semántico.
- ☐ Uso de selectores de CSS.
- ☐ Construir tu aplicación respetando el diseño realizado (maquetación).

**DOM**

- ☐ Uso de selectores del DOM.
- ☐ Manejo de eventos del DOM.
- ☐ Manipulación dinámica del DOM. (appendChild |createElement | createTextNode| innerHTML | textContent | etc.)

**JavaScript**

- ☐ Manipulación de strings.
- ☐ Uso de condicionales (if-else | switch | operador ternario)
- ☐ Uso de bucles (for | for..in | for..of | while)
- ☐ Uso de funciones (parámetros | argumentos | valor de retorno)
- ☐ Declaración correcta de variables (const & let)

**Testing**

- ☐ Testeo unitario.

**Estructura del código y guía de estilo**

- ☐ Organizar y dividir el código en módulos (Modularización)
- ☐ Uso de identificadores descriptivos (Nomenclatura | Semántica)
- ☐ Uso de linter (ESLINT)

**Git y GitHub**

- ☐ Uso de comandos de git (add | commit | pull | status |push)
- ☐ Manejo de repositorios de GitHub (clone | fork | gh-pages)

**UX**

- ☐ Diseñar la aplicación pensando y entendiendo al usuario.
- ☐ Crear prototipos para obtener feedback e iterar.
- ☐ Aplicar los principios de diseño visual (contraste, alineación, jerarquía)

## DATA LOVERS

**HTML y CSS**

- ☐ Uso de HTML semántico.
- ☐ Uso de selectores de CSS.
- ☐ Construir tu aplicación respetando el diseño realizado (maquetación).
- ☐ Uso de flexbox en CSS.

**DOM y Web APIs**

- ☐ Uso de selectores del DOM.
- ☐ Manejo de eventos del DOM.
- ☐ Manipulación dinámica del DOM. (appendChild |createElement | createTextNode| innerHTML | textContent | etc.)

**JavaScript**

- ☐ Uso de condicionales (if-else | switch | operador ternario)
- ☐ Uso de bucles (for | for..in | for..of | while)
- ☐ Uso de funciones (parámetros | argumentos | valor de retorno)
- ☐ Manipular arrays (filter | map | sort | reduce)
- ☐ Manipular objects (key | value)
- ☐ Uso ES modules (import | export)
- ☐ Diferenciar entre expression y statements.
- ☐ Diferenciar entre tipos de datos atómicos y estructurados.

**Testing**

- ☐ Testeo unitario.

**Estructura del código y guía de estilo**

- ☐ Organizar y dividir el código en módulos (Modularización)
- ☐ Uso de identificadores descriptivos (Nomenclatura | Semántica)
- ☐ Uso de linter (ESLINT)

**Git y GitHub**

- ☐ Uso de comandos de git (add | commit | pull | status |push)
- ☐ Manejo de repositorios de GitHub (clone | fork | gh-pages)
- ☐ Colaboración en Github (branches | pull requests | |tags)

**UX**

- ☐ Diseñar la aplicación pensando y entendiendo al usuario.
- ☐ Crear prototipos para obtener feedback e iterar.
- ☐ Aplicar los principios de diseño visual (contraste, alineación, jerarquía)
- ☐ Planear y ejecutar tests de usabilidad.

## SOCIAL NETWORK

**HTML y CSS**

- ☐ Uso de HTML semántico.
- ☐ Uso de selectores de CSS.
- ☐ Construir tu aplicación respetando el diseño realizado (maquetación).
- ☐ Uso de flexbox en CSS.

**DOM y Web APIs**

- ☐ Uso de selectores del DOM.
- ☐ Manejo de eventos del DOM.
- ☐ Manipulación dinámica del DOM. (appendChild |createElement | createTextNode| innerHTML | textContent | etc.)
- ☐ History API.
- ☐ localStorage.

**JavaScript**

- ☐ Uso de condicionales (if-else | switch | operador ternario)
- ☐ Uso de funciones (parámetros | argumentos | valor de retorno)
- ☐ Manipular arrays (filter | map | sort | reduce)
- ☐ Manipular objects (key | value)
- ☐ Uso ES modules (import | export)
- ☐ Diferenciar entre expression y statements.
- ☐ Diferenciar entre tipos de datos atómicos y estructurados.
- ☐ Uso de callbacks.
- ☐ Consumo de Promesas.

**Testing**

- ☐ Testeo unitario.
- ☐ Testeo asíncrono.
- ☐ Uso de librerias de Mock.

**Estructura del código y guía de estilo**

- ☐ Organizar y dividir el código en módulos (Modularización)
- ☐ Uso de identificadores descriptivos (Nomenclatura | Semántica)
- ☐ Uso de linter (ESLINT)

**Git y Github**

- ☐ Uso de comandos de git (add | commit | pull | status |push)
- ☐ Manejo de repositorios de GitHub (clone | fork | gh-pages)
- ☐ Colaboración en Github (branches | pull requests | |tags)
- ☐ Organización en Github (projects | issues | labels | milestones)

**Firebase**

- ☐ Firestore.
- ☐ Firebase Auth.
- ☐ Firebase security rules.
- ☐ Observadores. (onAuthStateChanged | onSnapshot)

**UX**

- ☐ Diseñar la aplicación pensando y entendiendo al usuario.
- ☐ Crear prototipos para obtener feedback e iterar.
- ☐ Aplicar los principios de diseño visual (contraste, alineación, jerarquía)
- ☐ Planear y ejecutar tests de usabilidad.

## BUGER QUEEN

**HTML y CSS**

- ☐ Uso de HTML semántico.
- ☐ Uso de selectores de CSS.
- ☐ Construir tu aplicación respetando el diseño realizado (maquetación).
- ☐ Uso de flexbox en CSS.
- ☐ Uso de Media Queries.

**JavaScript**

- ☐ Uso de condicionales (if-else | switch | operador ternario)
- ☐ Uso de funciones (parámetros | argumentos | valor de retorno)
- ☐ Manipular arrays (filter | map | sort | reduce)
- ☐ Manipular objects (key | value)
- ☐ Uso ES modules (import | export)
- ☐ Diferenciar entre expression y statements.
- ☐ Diferenciar entre tipos de datos atómicos y estructurados.
- ☐ Uso de callbacks.
- ☐ Consumo de Promesas.

**Testing**

- ☐ Testeo unitario.

**Estructura del código y guía de estilo**

- ☐ Organizar y dividir el código en módulos (Modularización)
- ☐ Uso de identificadores descriptivos (Nomenclatura | Semántica)
- ☐ Uso de linter (ESLINT)

**Git y Github**

- ☐ Uso de comandos de git (add | commit | pull | status | push)
- ☐ Manejo de repositorios de GitHub (clone | fork | gh-pages)
- ☐ Colaboración en Github (branches | pull requests | |tags)
- ☐ Organización en Github (projects | issues | labels | milestones)

**Firebase**

- ☐ Firestore.
- ☐ Firebase Auth.
- ☐ Firebase security rules.
- ☐ Observadores. (onAuthStateChanged | onSnapshot)

**Angular**

- ☐ Components & templates.
- ☐ Directivas estructurales (ngIf / ngFor)
- ☐ @Input | @Ouput
- ☐ Creación y uso de servicios.
- ☐ Manejos de rutas.
- ☐ Creación y uso Observables.
- ☐ Uso de HttpClient.
- ☐ Estilos de componentes (ngStyle / ngClass)

**React**

- ☐ JSX
- ☐ Componentes class y componentes function
- ☐ Props
- ☐ Eventos en React.
- ☐ Listas y keys.
- ☐ Renderizado condicional.
- ☐ Elevación de estados.
- ☐ Hooks
- ☐ CSS modules.
- ☐ React Router.

**UX**

- ☐ Diseñar la aplicación pensando y entendiendo al usuario.
- ☐ Crear prototipos para obtener feedback e iterar.
- ☐ Aplicar los principios de diseño visual (contraste, alineación, jerarquía)
- ☐ Planear y ejecutar tests de usabilidad.

## MD-LINKS

**JavaScript**

- ☐ Uso de condicionales (if-else | switch | operador ternario)
- ☐ Uso de funciones (parámetros | argumentos | valor de retorno)
- ☐ Manipular arrays (filter | map | sort | reduce)
- ☐ Manipular objects (key | value)
- ☐ Uso ES modules (import | export)
- ☐ Diferenciar entre expression y statements.
- ☐ Diferenciar entre tipos de datos atómicos y estructurados.
- ☐ Uso de callbacks.
- ☐ Consumo de Promesas.
- ☐ Creación de Promesas.

**Node**

- ☐ Uso de sistema de archivos. (fs, path)
- ☐ Instalar y usar módulos. (npm)
- ☐ Creación de modules. (CommonJS)
- ☐ Configuración de package.json.
- ☐ Configuración de npm-scripts
- ☐ Uso de CLI (Command Line Interface - Interfaz de Línea de Comando)

**Testing**

- ☐ Testeo unitario.
- ☐ Testeo asíncrono.
- ☐ Uso de librerias de Mock.
- ☐ Uso de Mocks manuales.
- ☐ Testeo para múltiples Sistemas Operativos.

**Estructura del código y guía de estilo**

- ☐ Organizar y dividir el código en módulos (Modularización)
- ☐ Uso de identificadores descriptivos (Nomenclatura | Semántica)
- ☐ Uso de linter (ESLINT)

**Git y GitHub**

- ☐ Uso de comandos de git (add | commit | pull | status |push)
- ☐ Manejo de repositorios de GitHub (clone | fork | gh-pages)
- ☐ Colaboración en Github (branches | pull requests | |tags)
- ☐ Organización en Github (projects | issues | labels | milestones)

**HTTP**

- ☐ Verbos HTTP (http.get)

**Fundamentos de programación**

- ☐ Recursión.