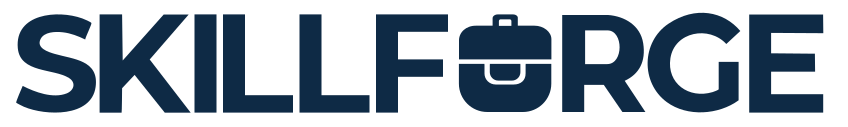
## 



**Solicitante:** I.T.S. - Instituto Tecnologico Superior Arias Balparda

**Nombre Fantasia Empresa:** SkillForge

**Grupo:** 3°ML

**Turno:** Vespertino

**Unidad curricular:** Programación FULL STACK

**Nombre de los integrantes del equipo:** Samuel Becerra, Sofia Hernández, Evelyn Guimaraens, Ivana Panizza (Referente)

**Fecha de entrega:** 28/07/2025

**Instituto Tecnológico Superior Arias Balparda.**

**Blvr. José Batlle y Ordóñez 3570 esq. Gral. Flores – Montevideo.**

Índice

[**1. Introducción 4**](#_n9p6u4g60i1b)

[1.1. Propósito 5](#_6cdb7ufq754p)

[1.2. Alcance 5](#_3x5h1e9vb27j)

[**2. Esquema Relacional 6**](#_3q00hfk7zo0g)

[2.1. Diagrama entidad relación (D.E.R ) 6](#_2vzvux61ltgo)

[2.2. Pasaje a tabla 7](#_qqqjz2weorbq)

[**3. WireFrames 8**](#_wo1gwx89r6gr)

[3.1. Pantalla de Inicio (Formato PC) 9](#_u8vbk3dvt6ys)

[3.2. Inicio de Sesión (Formato PC) 10](#_8cjgi78el1r5)

[3.3. Registro de Usuario (Formato PC) 10](#_c2jlu1th6e89)

[3.4. Información de Registro para el Proveedor(Formato PC) 11](#_45xzfq85jk6y)

[3.5. Pantalla de Inicio (Formato Mobile) 11](#_t69cqooaia1c)

[3.6. Inicio de Sesión y Registro de Usuario (Formato Mobile) 12](#_yipjdg1jtuma)

[3.7. Información de Registro para el Proveedor(Formato MOBILE) 12](#_qux8yl71k9n4)

[**4. Modelo-Vista-Controlador (MVC) 13**](#_8tvusvcbhruh)

[4.1. Modelos 15](#_iumcx8yje2zy)

[4.1.1. Estructura del Modelo Usuario 16](#_hvv134fkuar8)

[4.1.2. Estructura del Modelo Conexión con la Base de Datos (LocalHost) 18](#_sonn3dqy6nc6)

[4.2 Vistas 19](#_6eudgx274e8n)

[4.2.1. Estructura HTML del Formulario 20](#_8vma8swwua7c)

[4.2.2. Estructura HTML del LogIn 21](#_quqm1pkcvbcf)

[4.2.3. Estructura HTML del SignUp 22](#_qb7moal51khx)

[4.2.4. Estructura Script del Botón Mostrar/Ocultar Contraseña 24](#_gre9dxj1pxw0)

[4.2.5. Estructura Script de Coincidencia de Contraseña 25](#_m9ydhe22rzso)

[4.2.6. Estructura PHP de Post-SignUp Cliente 26](#_me9kw4vod3b6)

[4.2.7. Estructura de Post-SignUp Proveedor 27](#_n2uxkgbjck61)

[4.2.8. Estructura de Post-LogIn Cliente 28](#_stdh5tio7ycj)

[4.2.9. Estructura de Post-LogIn Proveedor 28](#_9rjgx4hnzu75)

[4.2.10. Estructura de Cierre de Sesión 29](#_ji6iz5ffm0qw)

[4.2.11. Estructura de Mensajes de Error 29](#_4shvafpvim7a)

[4.3. Controladores 30](#_idlpif1jdxl7)

[4.3.1. Estructura de Controlador Usuario 31](#_oz35froq36b8)

[**5. Interfaces de la Plataforma Web SkillForge 34**](#_p6n1glos2ij)

[5.1. Interfaz del Formulario LogIn - SignUp 34](#_vc59ealzo2f4)

[5.1.1. Interfaz de Pre/Post-SignUp de Usuario 36](#_w8h56n558m0f)

[5.1.1.1. Registro Cliente 36](#_snvgh0ds5au1)

[5.1.1.2. Registro Proveedor 37](#_qudkdixjkeuh)

[5.1.2. Interfaz Pre/Post-LogIn Usuario 38](#_n14dsunhsyay)

[5.1.2.1. LogIn Cliente 38](#_pvuln6x20z8h)

[5.1.2.2. LogIn Cliente 38](#_c16fbdhgo5ho)

[**Glosario 39**](#_vsxhkkszenfv)

[Vocabulario y simbología 39](#_z95ksvrubcad)

# 1. Introducción

SkillForge es una plataforma digital concebida con el propósito de facilitar el vínculo entre usuarios que requieren un oficio puntual y personas que ofrecen sus servicios profesionales de manera independiente. En esta etapa del desarrollo, se presenta la documentación correspondiente a la materia *Programación Full Stack*, centrada en aspectos técnicos clave del producto.

Este documento forma parte del proceso de construcción del producto de SkillForge, y tiene como objetivo documentar las decisiones adoptadas durante la implementación inicial del sistema. A través de un enfoque modular, se detallan los componentes que hacen posible la interacción entre usuarios, la gestión de datos y la estructura lógica del sistema.

A lo largo del presente documento, se incluyen los siguientes elementos:

* El diseño de la base de datos, incluyendo el modelo entidad-relación (DER) y su respectivo pasaje a tablas.
* La elaboración de wireframes, como referencia visual de las pantallas principales de la plataforma, tanto en formato PC como Mobile.
* La definición de la arquitectura interna del sistema, basada en el patrón Modelo-Vista-Controlador (MVC).
* El desarrollo de los modelos, vistas y controladores que permiten el registro, inicio de sesión y diferenciación de perfiles según el tipo de usuario.
* La implementación del sistema de autenticación (LogIn y SignUp), incluyendo validaciones de formularios y manejo de sesiones.
* Una descripción general de las interfaces funcionales y flujos posteriores al registro o inicio de sesión.
* Un glosario técnico que recopila términos clave, siglas y sintaxis utilizadas a lo largo del proyecto.

Este documento se construyó respetando la lógica y requerimientos del proyecto institucional de SkillForge, y está alineado con los lineamientos de la materia, sirviendo como evidencia del desarrollo progresivo de la solución tecnológica.

## 1.1. Propósito

El objetivo principal de SkillForge como empresa es desarrollar una plataforma digital que permita forjar vínculos laborales entre quienes necesitan un oficio puntual y quienes poseen las habilidades para brindarlo. Inspirados en modelos reconocidos como MercadoLibre o PedidosYa, buscamos adaptar esa lógica al mundo de los oficios locales, priorizando la confianza, la reputación y la organización.

Desde una perspectiva empresarial, este documento representa la especificación de requerimientos del producto central de SkillForge, sirviendo como hoja de ruta para su desarrollo, evaluación y evolución futura. El objetivo es asegurar una base sólida y escalable para ofrecer una experiencia óptima a usuarios finales, clientes potenciales e inversores.

## 1.2. Alcance

Este documento abarca la especificación funcional y técnica del producto digital de SkillForge: una plataforma de contratación de oficios accesible desde múltiples dispositivos. Contempla el relevamiento de necesidades del mercado, la identificación de actores clave, el diseño de funcionalidades principales, y la planificación de futuras ampliaciones.

En esta primera versión, el desarrollo se centrará en funcionalidades esenciales como registro de usuarios, autenticación, creación de perfiles diferenciados, búsqueda de servicios, sistema de reputación, mensajería interna y administración de cuentas. Funcionalidades como la integración con pasarelas de pago o la validación automática de antecedentes se consideran como mejoras a integrar en siguientes etapas del roadmap.

Este alcance responde tanto a las capacidades actuales de la empresa como al objetivo de entregar un producto mínimo viable que permita validar hipótesis de negocio, captar usuarios y demostrar el potencial de la plataforma a interesados estratégicos.

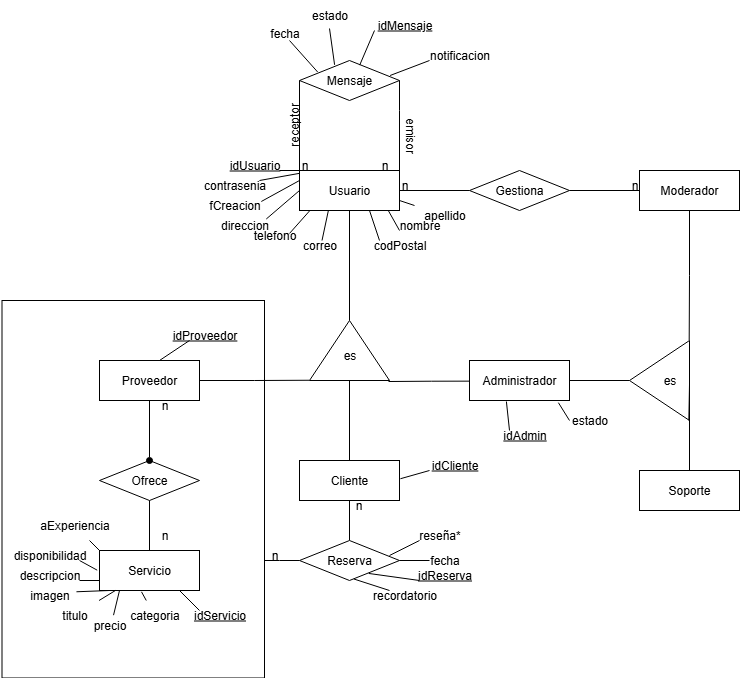
# 2. Esquema Relacional

# 

En SkillForge, durante esta etapa del proyecto, llevamos a cabo el diseño del modelo de datos, el cual establece la forma en que se organizará y almacenará la información. Este proceso se basó en el Diagrama Entidad-Relación (DER) previamente elaborado. A partir de dicho esquema, desarrollamos un conjunto estructurado de tablas que nos permitió representar de manera clara las entidades del sistema y definir con precisión las relaciones entre ellas, asegurando así una base de datos coherente y eficiente.

# 

## 2.1. Diagrama entidad relación (D.E.R )

****

## 2.2. **Pasaje a tabla**

Con el objetivo de facilitar la comprensión de la estructura de la base de datos, en el pasaje del Diagrama Entidad-Relación al modelo relacional, se incorporó un sistema de codificación por colores. En este esquema, las claves primarias y foráneas se distinguen visualmente mediante colores específicos, lo cual permite identificar de forma clara y rápida las relaciones entre las distintas tablas, mejorando así la interpretación del modelo.

Primary Key:AZUL

Foreign Key:VERDE

**Entidades:**

* Usuario (IdUsuario, nombre, telefono, correo, fechaCreacion)
* Cliente (idUsuario, idCliente)
* Proveedor (idUsuarioProv, idProveedor)
* Administrador (IdAdmin, idUsuarioAdmin, estado)
* Moderador (idAdmin, idUsuarioAdmin)
* Soporte (idAdmin, idUsuarioAdmin)
* Servicio (idServicio, disponibilidad, categoria, descripcion, precio, titulo, imagen, aExperiencia)

**Relaciones:**

* Mensaje (idMensaje, receptor, emisor, fecha, notificacion, estado)
* Gestiona (IdUsuario, IdAdmin)
* Ofrece (idServicio, IdProveedor)
* Reserva (idReserva, IdCliente, idServicio, IdProveedor, fecha, recordatorio)

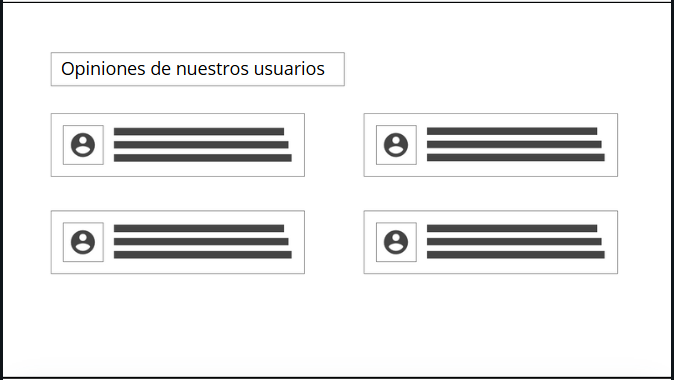
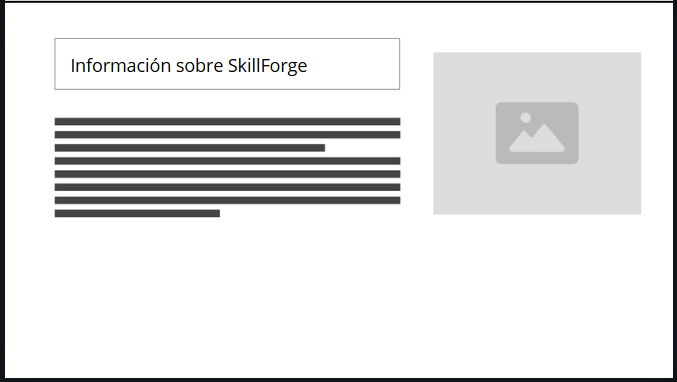
**Tablas derivadas de atributos multivaluados:**

* Reserva\_Reseña (IdReserva, reseña)

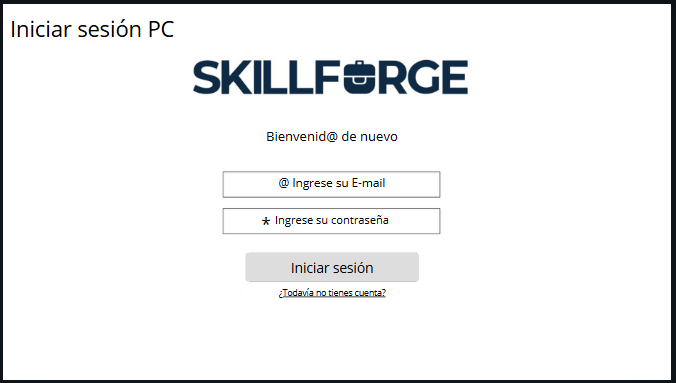
# 3. WireFrames

En SkillForge, como parte del proceso de diseño de la interfaz, desarrollamos un wireframe que nos permitió visualizar la estructura general de la plataforma antes de avanzar con el diseño final. Este prototipo funcional nos sirvió para definir la distribución de los elementos en pantalla, jerarquizar la información y validar la experiencia del usuario.

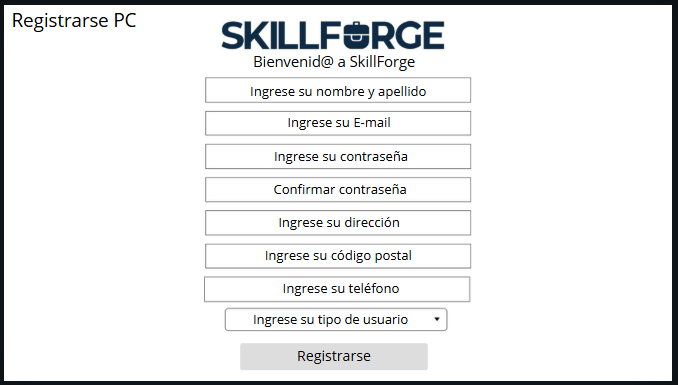
## 3.1. Pantalla de Inicio (Formato PC)



## 3.2. Inicio de Sesión (Formato PC)



## 3.3. Registro de Usuario (Formato PC)

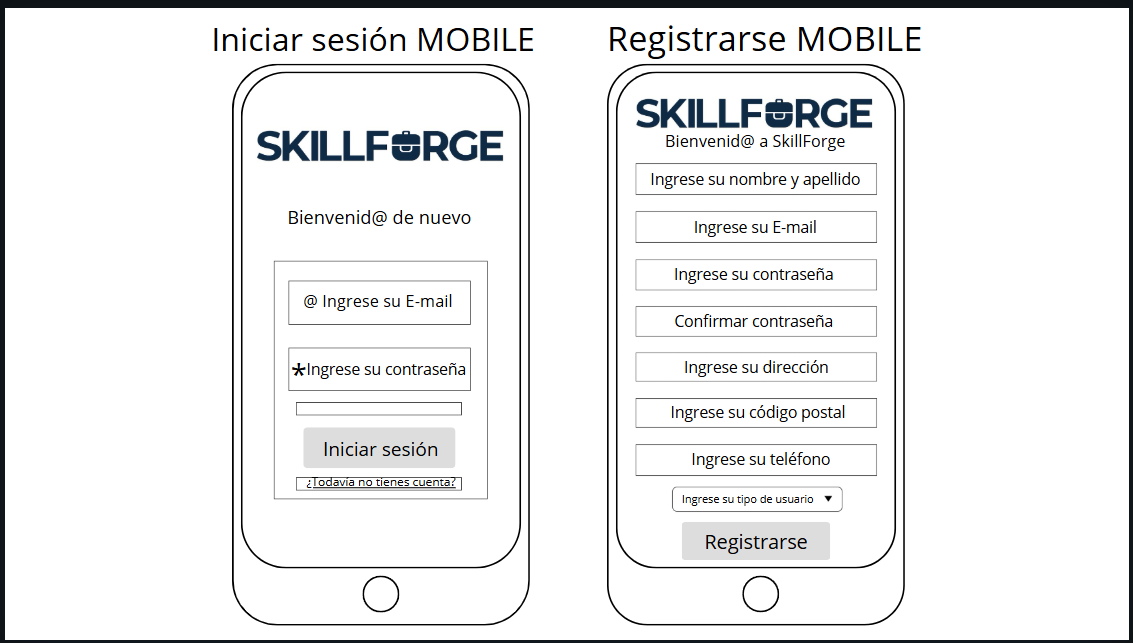


## 3.4. Información de Registro para el Proveedor(Formato PC)



## 3.5. Pantalla de Inicio (Formato Mobile)

## 3.6. Inicio de Sesión y Registro de Usuario (Formato Mobile)



## 3.7. Información de Registro para el Proveedor(Formato MOBILE)

# 4. Modelo-Vista-Controlador (MVC)

SkillForge ha adoptado la arquitectura Modelo-Vista-Controlador (MVC) como estándar para la estructuración del código fuente, asegurando así una separación clara entre la lógica de negocio, la presentación y la manipulación de datos. Esta decisión estratégica permite escalar el proyecto de forma eficiente, facilitar el mantenimiento y fomentar el trabajo colaborativo entre equipos técnicos.

**Estructura de Carpetas**

La estructura base del proyecto está organizada de la siguiente manera:

/Empresa SkillForge

│

├── controladores/

│ └── controllerUsuario.php

│

├── modelos/

│ ├── modeloUsuario.php

│ └── conexionBD.php

│

├── vistas/

│ ├── dashboardCliente.php

│ ├── dashboardProveedor.php

│ ├── perfilCliente.php

│ ├── perfilProveedor.php

│ ├── indexLogin.html

│ ├── indexLogin.php

│ └── logout.php

│

├── recursos/

│ ├── css/

│ ├── js/

│ └── img/

│ ├── logoskillforgeoscuro.png

│ └── logoskillforgeoscuro.ico

Cada carpeta cumple un rol específico:

* /modelos/: Contiene la lógica de acceso a datos y conexión a la base de datos.
* /controladores/: Recibe peticiones del usuario y coordina acciones entre modelo y vista.
* /vistas/: Presenta la interfaz visual al usuario, incluyendo formularios, menús y paneles.
* /recursos/: Contiene archivos estáticos reutilizables (imágenes, estilos, íconos y scripts JS).

## 4.1. Modelos

Los modelos en SkillForge representan los componentes encargados de gestionar y manipular los datos internos de la aplicación, incluyendo la conexión con la base de datos, la definición de estructuras de datos, y la implementación de métodos que permiten realizar operaciones CRUD (crear, leer, actualizar, eliminar).

Desde la perspectiva de ingeniería de software empresarial, estos modelos son fundamentales para asegurar la integridad de los datos y facilitar un desarrollo seguro y estructurado. Actualmente, se han desarrollado dos modelos esenciales:

* **modeloUsuario.php**: contiene la clase Usuario, que gestiona el registro, la verificación de credenciales y la validación de email.
* **conexionBD.php**: se encarga de establecer una conexión eficiente y segura con la base de datos local MySQL, utilizando la clase ConexionBD.

Estos modelos han sido diseñados para que sean reutilizables y modulares, permitiendo ser invocados desde cualquier controlador sin duplicar lógica de negocio.

### 4.1.1. Estructura del Modelo Usuario

| **<!--Inicio código de modeloUsuario.php-->**  **<?php** **require\_once**(**\_\_DIR\_\_** . '/../modelos/conexionBD.php'); **class** **Usuario** {   **private** $conn;  **private** $nombre;  **private** $email;  **private** $pass;  **private** $pass2;  **private** $direccion;  **private** $codPostal;  **private** $telefono;  **private** $tipoUsuario;  **public** **function** **setDatosUsuario**($nombre, $email, $pass, $direccion, $codPostal, $telefono, $tipoUsuario)  {   **$this**->nombre=$nombre;  **$this**->email=$email;  **$this**->pass=$pass;  **$this**->direccion=$direccion;  **$this**->codPostal=$codPostal;  **$this**->telefono=$telefono;  **$this**->tipoUsuario=$tipoUsuario;  }  **public** **function** **\_\_construct**()   {  $conexion = **new** ConexionBD();  **$this**->conn = $conexion->obtenerConexion();  }  **public** **function** **registrarUsuario**($nombre, $email, $pass, $direccion, $codPostal, $telefono, $tipoUsuario)   {  $sql = "INSERT INTO usuarios (nombre, email, contrasenia, direccion, cod\_postal, telefono, tipo\_usuario)   VALUES (?, ?, ?, ?, ?, ?, ?)";  $stmt = **$this**->conn->prepare($sql);  $encriptPass = password\_hash($pass, PASSWORD\_DEFAULT);  $stmt->bind\_param("sssssss", $nombre, $email, $encriptPass, $direccion, $codPostal, $telefono, $tipoUsuario);  **return** $stmt->execute();  }   **public** **function** **verificarEmailExistente**($email)   {  $sql = "SELECT COUNT(\*) as total FROM usuarios WHERE email = ?";  $stmt = **$this**->conn->prepare($sql);  $stmt->bind\_param("s", $email);  $stmt->execute();  $resultado = $stmt->get\_result()->fetch\_assoc();  **return** $resultado['total'] > 0;  }   **public** **function** **verificarLogin**($email, $contraseniaIngresada)   {   $sql = "SELECT \* FROM usuarios WHERE email = ?";  $stmt = **$this**->conn->prepare($sql);  $stmt->bind\_param("s", $email);  $stmt->execute();  $resultado = $stmt->get\_result();  **if** ($resultado->num\_rows === 1)   {  $usuario = $resultado->fetch\_assoc();  **if** (password\_verify($contraseniaIngresada, $usuario['contrasenia'])) {  **return** $usuario;  }  }  **return** **false**;  } } **?>**  **<!--Fin código de modeloUsuario.php-->** |
| --- |

## 

### 4.1.2. Estructura del Modelo Conexión con la Base de Datos (LocalHost)

| **<!--Inicio código de conexionBD.php-->**  **<?php** **class** **ConexionBD** {  **private** $servidor = "localhost"; **private** $usuario = "root"; **private** $contrasenia = ""; **private** $baseDeDatos = "empresaskillforge"; **private** $conexion;   **public** **function** **obtenerConexion**() {  **$this**->conexion = **new** mysqli(**$this**->servidor, **$this**->usuario, **$this**->contrasenia, **$this**->baseDeDatos);  **if** (**$this**->conexion->connect\_error)  {  **exit**("Error de conexión: " . **$this**->conexion->connect\_error);  }  **return** **$this**->conexion;  } } **?>**  **<!--Fin código de conexionBD.php-->** |
| --- |

## 

## 4.2 Vistas

Las vistas en SkillForge constituyen la capa encargada de presentar la información al usuario final. En este proyecto, incluyen tanto el diseño estático en HTML como los scripts dinámicos que permiten validar formularios, mostrar/ocultar contraseñas y gestionar la interacción visual del usuario con la plataforma.

Desde la óptica empresarial, el diseño y funcionalidad de las vistas son claves para ofrecer una experiencia de usuario intuitiva, accesible y profesional. Las vistas están alojadas en la carpeta vistas/ y están estructuradas para atender distintos momentos del flujo de uso:

* **indexLogin.html / indexLogin.php:** formulario de inicio de sesión y registro de usuario.
* **perfilCliente.php / perfilProveedor.php:** formularios post-registro donde se completan datos adicionales según el tipo de usuario.
* **dashboardCliente.php / dashboardProveedor.php:** interfaces de bienvenida luego del inicio de sesión exitoso.
* **logout.php:** gestiona el cierre de sesión.

Se complementan con scripts de validación de contraseñas y manipulación de campos visibles, mejorando la seguridad y usabilidad del formulario. Esto responde al objetivo de SkillForge de garantizar una experiencia funcional y atractiva desde el primer contacto del usuario con la plataforma.

### 4.2.1. Estructura HTML del Formulario

| **<!--Inicio código de indexLogin.html-->**  **<!DOCTYPE html>** <html lang="es"> <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>  <link rel="icon" href="" type="/EMPRESA%20SKILLFORGE/recursos/img/logoskillforgeoscuro.ico">  <link rel="shortcut icon" href="/EMPRESA%20SKILLFORGE/recursos/img/logoskillforgeoscuro.ico" type="image/x-icon">  <title>SkillForge</title> </head> 4.2.2. Estructura HTML del LogIn **<!--Incio de Body-->**  <body>   <fieldset>  <legend>Iniciar Sesión</legend>  <form action="/EMPRESA%20SKILLFORGE/controladores/controllerUsuario.php" method="post">  <input type="hidden" name="accion" value="loginUsuario">  <label for="emailUsuario">Email</label><br>  <input type="text" name="emailUsuario" id="emailUsuario" required><br><br>  <label for="contraseniaUsuario"><legend>Contraseña</legend></label>  <input type="password" id="logcontraseniaUsuario" name="contraseniaUsuario" minlength="8" maxlength="16" required>  <button type="button" class="password" data-target="logcontraseniaUsuario"><i class="fa fa-eye-slash"></i></button><br><br>  <input type="submit" value="Iniciar Sesión">  </form>  </fieldset> 4.2.3. Estructura HTML del SignUp <fieldset>   <legend>Registrar Usuario</legend>  <form id="formRegistro" action="/EMPRESA%20SKILLFORGE/controladores/controllerUsuario.php" method="post">  <input type="hidden" name="accion" value="registrarUsuario">  <label for="nombreUsuario"><legend>Nombre Completo</legend></label>  <input type="text" id="nombreUsuario" name="nombreUsuario" placeholder="Nombre Apellido" required><br><br>  <label for="emailUsuario"><legend>Email</legend></label>  <input type="text" id="emailUsuario" name="emailUsuario" placeholder="tucorreo@ejemplo.com" required><br><br>  <label for="contraseniaUsuario"><legend>Contraseña</legend></label>  <input type="password" title="Las contraseñas deben coincidir" id="regcontraseniaUsuario" name="contraseniaUsuario" placeholder="Mínimo 8 caracteres" minlength="8" maxlength="16" required>  <button type="button" class="password" data-target="regcontraseniaUsuario"><i class="fa fa-eye-slash"></i></button><br><br>  <label for="vfcontraseniaUsuario"><legend>Verificar Contraseña</legend></label>  <input type="password" id="regvfcontraseniaUsuario" name="vfcontraseniaUsuario" placeholder="Repetir contraseña" minlength="8" maxlength="16" required>  <button type="button" class="password" data-target="regvfcontraseniaUsuario"><i class="fa fa-eye-slash"></i></button><br><br>  <label for="direccionUsuario"><legend>Dirección</legend></label>  <input type="text" id="direccionUsuario" name="direccionUsuario" placeholder="Av. Calle 1234" required><br><br>  <label for="codPostalUsuario"><legend>Código Postal</legend></label>  <input type="text" id="codPostalUsuario" name="codPostalUsuario" placeholder="00000" pattern="\d{5}" maxlength="5" required><br><br>  <label for="telefonoUsuario"><legend>Teléfono</legend></label>  <input type="text" id="telefonoUsuario" name="telefonoUsuario" placeholder="09#######" pattern="09\d{7}" maxlength="9" required><br><br>  <label for="tipoUsuario"><legend>Tipo de Usuario</legend></label>  <select name="tipoUsuario" id="tipoUsuario" required>  <option value="">--Seleccione una opción--</option>  <option value="cliente">Cliente</option>  <option value="proveedor">Proveedor</option>  </select><br><br>  <input type="submit" value="CREAR CUENTA">  </form>  </fieldset> </body> 4.2.4. Estructura Script del Botón Mostrar/Ocultar Contraseña  <script>  $(document).ready(function () {  $('.password').click(function () {  const inputId = $(this).data('target');  const input = $('#' + inputId);  const icon = $(this).find('i');  if (input.attr('type') === 'password') {  input.attr('type', 'text');  icon.removeClass('fa-eye-slash').addClass('fa-eye');  } else {  input.attr('type', 'password');  icon.removeClass('fa-eye').addClass('fa-eye-slash');  }  });  }); </script> 4.2.5. Estructura Script de Coincidencia de Contraseña <script>  document.addEventListener("DOMContentLoaded", function () {  const form =document.getElementById("formRegistro");  const pass1 =document.getElementById("contraseniaUsuario");  const pass2 =document.getElementById("vfcontraseniaUsuario");  pass2.addEventListener("input", function () {  if (pass1.value !== pass2.value) {  pass2.setCustomValidity("Las contraseñas no coinciden.");  } else {  pass2.setCustomValidity("");}  });  form.addEventListener("submit", function (e) {  if (pass1.value !== pass2.value) {  pass2.setCustomValidity("Las contraseñas no coinciden.");  pass2.reportValidity();  e.preventDefault();  } else {  pass2.setCustomValidity("");}  });  }); </script> </html> **<!--Fin código de indexLogin.html-->** |
| --- |
|  |
|  |

### 4.2.6. Estructura PHP de Post-SignUp Cliente

| **<!--Inicio código de perfilCliente.php-->**  **<?php** $email = $\_GET['email'] ?? ''; **?>** **<!DOCTYPE html>** <html lang="es"> <head>  <meta charset="UTF-8">  <title>Perfil Cliente - SkillForge</title> </head> <body>  <h2>Registro exitoso</h2>  <p>¡Bienvenido a SkillForge, estimado cliente!</p>  <p>Ahora completá la información de tu perfil para comenzar a usar la plataforma.</p>  <form action="guardarPerfilCliente.php" method="post">  <input type="hidden" name="email" value="<?= htmlspecialchars($email) ?>">  <label for="edad">Edad:</label>  <input type="number" name="edad" id="edad" min="18" max="100"><br><br>  <label for="genero">Género:</label>  <select name="genero" id="genero">  <option value="">--Seleccionar--</option>  <option value="masculino">Masculino</option>  <option value="femenino">Femenino</option>  <option value="otro">Otro</option>  </select><br><br>  <input type="submit" value="Guardar Perfil">  <input href="../vistas/logout.php" type="submit" value="Volver al Menu">  </form> </body> </html>**<!--Fin código de perfilCliente.php-->** |
| --- |

# 

### 4.2.7. Estructura de Post-SignUp Proveedor

| **<!--Inicio código de perfilProveedor.php-->**  **<?php** $email = $\_GET['email'] ?? ''; **?>** **<!DOCTYPE html>** <html lang="es"> <head>  <meta charset="UTF-8">  <title>Perfil Proveedor - SkillForge</title> </head>  <body>  <h2>Registro exitoso</h2>  <p>¡Bienvenido a SkillForge, proveedor!</p>  <p>Por favor completá los datos de tu perfil profesional:</p>  <form action="guardarPerfilProveedor.php" method="post">  <input type="hidden" name="email" value="<?= htmlspecialchars($email) ?>">  <label for="oficio">Seleccioná tu oficio principal:</label>  <select name="oficio" id="oficio" required>  <option value="">--Seleccionar--</option>  <option value="carpintero">Carpintero</option>  <option value="desarrollador\_web">Desarrollador Web</option>  <option value="electricista">Electricista</option>  <option value="mecanico">Mecánico</option>  <option value="plomero">Plomero</option>  </select><br><br>  <label for="experiencia">Años de experiencia:</label>  <input type="number" name="experiencia" id="experiencia" min="0" max="50"><br><br>  <label for="descripcion">Descripción profesional:</label><br>  <textarea name="descripcion" id="descripcion" cols="30" rows="5" placeholder="Contanos sobre vos y tus servicios..."></textarea><br><br>  <input type="submit" value="Guardar Perfil">  <input href="../vistas/logout.php" type="submit" value="Volver al Menu">  </form> </body> </html> **<!--Fin código de perfilProveedor.php-->** |
| --- |

### 

### 4.2.8. Estructura de Post-LogIn Cliente

| **<!--Inicio código de dashboarCliente.php-->**  **<?php** session\_start(); **if** (!**isset**($\_SESSION['usuario']) || $\_SESSION['usuario']['tipo\_usuario'] !== 'cliente') {  header("Location:indexLogin.php");  **exit**(); } **?>** <h1>Bienvenido, Cliente</h1> <p>Has iniciado sesión correctamente.</p> <a href="logout.php">Cerrar sesión</a>  **<!--Final código de dashboarCliente.php-->** |
| --- |

## 

### 4.2.9. Estructura de Post-LogIn Proveedor

| **<!--Inicio código de dashboarProveedor.php-->**  **<?php** session\_start(); **if** (!**isset**($\_SESSION['usuario']) || $\_SESSION['usuario']['tipo\_usuario'] !== 'proveedor') {  header("Location:indexLogin.php");  **exit**(); } **?>** <h1>Bienvenido, Proveedor</h1> <p>Has iniciado sesión correctamente.</p> <a href="logout.php">Cerrar sesión</a>  **<!--Fin código de dashboarProveedor.php-->** |
| --- |

# 

### 4.2.10. Estructura de Cierre de Sesión

| **<?php** session\_start(); session\_unset(); session\_destroy(); header("Location: ../vistas/indexLogin.html?logout=true"); **exit**(); **?>** |
| --- |

### 4.2.11. Estructura de Mensajes de Error

| **<?php** **if** (**isset**($\_GET['error'])): **?>**  <p style="color:red;">  **<?php**   **switch**($\_GET['error']) {  **case** 'Campos\_Vacios': **echo** "Por favor, completá todos los campos."; **break**;  **case** 'Passwords\_no\_coinciden': **echo** "Las contraseñas no coinciden."; **break**;  **case** 'Email\_ya\_registrado': **echo** "Este email ya está registrado."; **break**;  **case** 'Error\_al\_registrar': **echo** "Error al registrar. Intente nuevamente."; **break**;  }  **?>**  </p> **<?php** **endif**; **?>** |
| --- |

## 

## 4.3. Controladores

Los controladores son responsables de orquestar la lógica entre las vistas y los modelos. En SkillForge, cada acción que realiza un usuario, como registrarse o iniciar sesión, es procesada por un controlador que valida la información, la procesa mediante los modelos y devuelve una respuesta adecuada a la vista correspondiente.

Actualmente se utiliza un controlador principal:

* **controllerUsuario.php**: gestiona todas las solicitudes POST relacionadas al registro y autenticación de usuarios. Implementa validaciones, redirecciones condicionales y lógica de manejo de errores.

Desde el enfoque profesional y de calidad de software, los controladores han sido diseñados siguiendo principios de claridad, responsabilidad única y seguridad, asegurando que los datos del usuario se manejen de forma estructurada y protegida en todo momento.

### 4.3.1. Estructura de Controlador Usuario

| **<?php**  **require\_once**(**\_\_DIR\_\_** . '/../modelos/modeloUsuario.php');  **if**($\_SERVER['REQUEST\_METHOD']==='POST') {  $accion = $\_POST['accion'];  $usuario = **new** Usuario();  **switch**($accion)  {  **case** "registrarUsuario":  $nombre = $\_POST['nombreUsuario'];  $email = $\_POST['emailUsuario'];  $pass = $\_POST['contraseniaUsuario'];  $pass2 = $\_POST['vfcontraseniaUsuario'];  $direccion = $\_POST['direccionUsuario'];  $codPostal = $\_POST['codPostalUsuario'];  $telefono = $\_POST['telefonoUsuario'];  $tipoUsuario = $\_POST['tipoUsuario'];  **if** (**empty**($nombre) || **empty**($email) || **empty**($pass) || **empty**($pass2) || **empty**($direccion) || **empty**($codPostal) || **empty**($telefono))   {  header("Location: ../vistas/indexLogin.php?error=Campos\_Vacios");  **exit**();   }  **if** ($pass !== $pass2) {  header("Location: ../vistas/indexLogin.php?error=Passwords\_no\_coinciden");  **exit**();  }  **if** ($usuario->verificarEmailExistente($email)) {  header("Location: ../vistas/indexLogin.php?error=Email\_ya\_registrado");  **exit**();  }  **if** ($usuario->registrarUsuario($nombre, $email, $pass, $direccion, $codPostal, $telefono, $tipoUsuario)) {    **if** ($tipoUsuario === 'cliente') {  header("Location: ../vistas/perfilCliente.php?email=$email");  } **elseif** ($tipoUsuario === 'proveedor') {  header("Location: ../vistas/perfilProveedor.php?email=$email");  }  **exit**();  } **else** {  header("Location: ../vistas/indexLogin.php?error=Error\_al\_registrar");  **exit**();  }  **break**;   **case** "loginUsuario":  session\_start();   $email = $\_POST['emailUsuario'];  $pass = $\_POST['contraseniaUsuario'];   $datosUsuario = $usuario->verificarLogin($email, $pass);   **if** ($datosUsuario) {  $\_SESSION['usuario'] = $datosUsuario;  **if** ($datosUsuario['tipo\_usuario'] === 'cliente') {  header("Location: ../vistas/dashboardCliente.php");  }  **elseif** ($datosUsuario['tipo\_usuario'] === 'proveedor') {  header("Location: ../vistas/dashboardProveedor.php");  } **else** {  header("Location: ../vistas/dashboard.php");  }  } **else** {  header("Location: ../vistas/indexLogin.php?error=Credenciales\_invalidas");  }  **exit**();  **break**;  } } **?>** |
| --- |

# 5. Interfaces de la Plataforma Web SkillForge

La interfaz de usuario (UI) es un componente clave del producto SkillForge. Representa la manera en que los usuarios interactúan con la plataforma, y como empresa, hemos priorizado una experiencia fluida, accesible y moderna. Las interfaces fueron diseñadas utilizando un enfoque responsivo que garantiza una correcta visualización tanto en dispositivos móviles como en computadoras de escritorio (PC).

Se han implementado estructuras diferenciadas para cada etapa del ciclo de usuario: registro, inicio de sesión y navegación inicial. Esto permite una interacción intuitiva, que reduce la curva de aprendizaje del usuario final y facilita su adopción. Las interfaces están fundamentadas en los wireframes diseñados previamente, respetando los principios de usabilidad y estética minimalista.

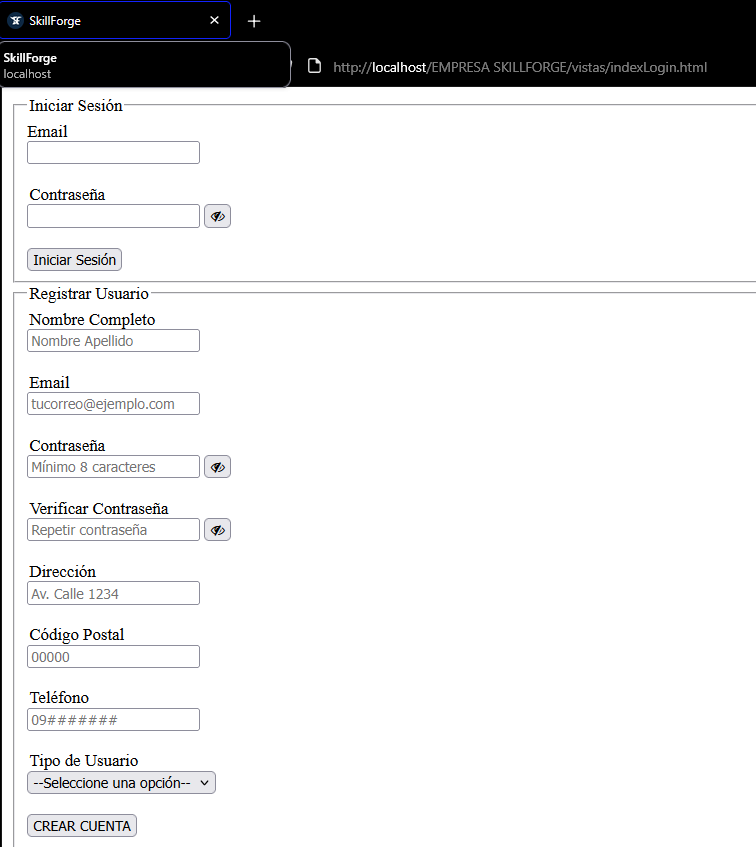
## 5.1. Interfaz del Formulario LogIn - SignUp

La interfaz principal de la plataforma incluye un formulario de autenticación que permite al usuario iniciar sesión o registrarse, según corresponda. Esta sección es crítica ya que representa el primer punto de contacto entre la empresa y sus clientes o proveedores.

El diseño incorpora los siguientes elementos funcionales:

* Formularios claros y breves para facilitar el ingreso de datos.
* Validación de contraseñas en tiempo real para garantizar seguridad.
* Selección del tipo de usuario (cliente o proveedor) para una navegación adaptada.
* Iconografía intuitiva para acciones como mostrar/ocultar contraseña.

Esta interfaz fue desarrollada considerando los estándares de accesibilidad y compatibilidad multiplataforma.



### 

### 5.1.1. Interfaz de Pre/Post-SignUp de Usuario

Una vez registrado el usuario, el sistema redirige automáticamente a una pantalla de creación de perfil personalizada, dependiendo del tipo de usuario seleccionado. Esta separación es fundamental para brindar una experiencia especializada desde el inicio y fortalecer la identidad de cada perfil dentro de la plataforma.

## 

#### 5.1.1.1. Registro Cliente

El usuario tipo cliente es guiado hacia un formulario donde debe ingresar información complementaria como edad y género. Esta información permite a la plataforma ofrecer recomendaciones más personalizadas y, en etapas futuras, filtrar servicios más acordes al perfil del cliente.

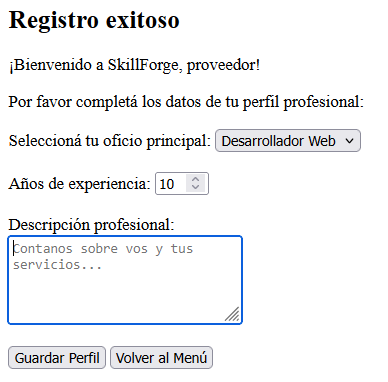
#### 

#### 

#### 5.1.1.2. Registro Proveedor

En el caso de los proveedores, el formulario posterior al registro incluye campos orientados a definir el perfil profesional: oficio principal, experiencia y descripción de servicios. Esta información es crítica para construir un perfil sólido dentro de SkillForge, ya que será la base para que los clientes puedan elegir entre diferentes prestadores.





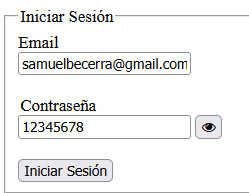
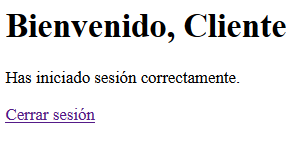
### 

### 5.1.2. Interfaz Pre/Post-LogIn Usuario

Una vez que el usuario inicia sesión exitosamente, es redirigido automáticamente a su dashboard personalizado, donde podrá visualizar sus opciones, servicios disponibles y notificaciones correspondientes.

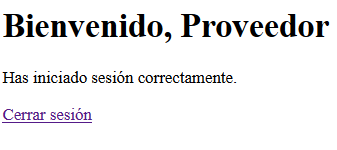
#### 5.1.2.1. LogIn Cliente

El cliente autenticado accede a una pantalla de bienvenida con su información de perfil y las primeras funcionalidades básicas de la plataforma. En esta etapa, el diseño busca generar familiaridad y confianza.

#### 5.1.2.2. LogIn Cliente

El proveedor, al iniciar sesión, accede a su panel profesional. Allí podrá comenzar a gestionar su oferta de servicios, actualizar su perfil y, en futuras versiones, visualizar métricas de interacción y mensajes de clientes interesados

# Glosario

## Vocabulario y simbología

A continuación se listan las abreviaciones, palabras clave y símbolos técnicos utilizados en el presente documento, tanto en los títulos como en el código fuente incluido:

**AUTO\_INCREMENT**: Traducción al español “autoincremental”.

**CREATE**: Traducción al español “crear”.

**CRUD**: Create, Read, Update, Delete: operaciones básicas sobre una base de datos.

**DATABASE**: Traducción al español “Base de datos”.

**DB o DATABASE:** Base de datos es la estructura donde se almacenan los datos del sistema.

**exit()**: Instrucción que finaliza la ejecución del script actual.

**FOREIGN KEY:** Traducción al español “clave foránea” es el campo que referencia a una clave primaria en otra tabla.

**GET / POST:** Métodos HTTP utilizados para enviar datos desde o hacia el servidor.

**HTML:** HyperText Markup Language, traducido al español lenguaje de marcado utilizado en la web.

**href / src**: Atributos de HTML usados para enlazar archivos o recursos.

**ICO**: Archivo de ícono utilizado por navegadores para representar visualmente una pestaña.

**input / label / select**: Elementos HTML utilizados en formularios.

**INSERT**: Traducción al español “insert”.

**into**: Traducción al español “dentro”.

**isset()**: Función PHP que determina si una variable está definida.

**JS**: JavaScript es un lenguaje de programación para la interacción del lado del cliente.

**MVC**: Modelo-Vista-Controlador es el patrón de diseño que separa lógica de presentación y negocio.

**NOT NULL**: Traducción al español “no nulo”.

**PC**: Traducción al español “ordenador personal o computadora”.

**PHP**: Hypertext Preprocessor es el lenguaje de programación del lado del servidor.

**PRIMARY KEY**: Traducción al español “clave primaria” es el campo único que identifica un registro.

**REFERENCES**: Traducción al español “referencias”.

**require\_once()**: Instrucción de PHP para incluir archivos solo una vez en la ejecución.

**SELECT**: Traducción al español “seleccione”.

**Session**: Sesión de usuario: conjunto de datos temporales asociados a un usuario autenticado.

**SQL**: Structured Query Language es el lenguaje para gestionar bases de datos.

**TABLE**: Traducción al español “tabla”.

**USE**: Traducción al español “usar”.

**VALUES**: Traducción al español “valores”.

**WireFrame**: Esquema visual de una aplicación web antes de ser programada.

**MOBILE**: Traducción al español “Móvil o Teléfono”.

**header()**: Función PHP para redireccionar a otra URL o modificar cabeceras.

**Hoja testigo:**

**Firma del Profesor**