3-8-2016

Yording Cala Sánchez

DOCUMENTO DE ARQUITECTURA SOFTWARE

ClubMoney

CONTENIDO

# INTRODUCCIÓN

El objetivo de este documento es documentar la arquitectura de software para el desarrollo de la aplicación MoneyClub

# DESCRIPCIÓN DE LA SOLUCIÓN

Actualmente las sociedades de ahorro colectivo conocidas comúnmente como “natilleras” no poseen un sistema el cual les permita facilitar el manejo del dinero, aportes de los integrantes y el préstamo a terceros e integrantes. Se necesita migrar todo el manejo de ahorros realizado en plantillas físicas o plantillas en Excel a un sistema que permita mostrar el estado de la natillera en tiempo real, agendar eventos o reuniones y poder realizar préstamos a terceros mucho mas seguros.

# REQUISITOS QUE ORIENTAN LA ARQUITECTURA

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 1** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de Roles | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador crear un nuevo rol en la aplicación.  Para crear un nuevo rol se requiere como mínimo la siguiente información:   * id (auto numérico) * rol * descripción   **Funciones**   * Validaciones (). * Crear () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 2** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar roles | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar un rol de la aplicación.  Para eliminar un rol se requiere como mínimo la siguiente información:   * Id.   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina el rol de ser estrictamente necesario de lo contrario se inhabilita. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 3** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de los roles. | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de un rol en la aplicación.  Para actualizar un rol se requiere como mínimo la siguiente información:   * id * rol * descripción   **Funciones**   * Validaciones (). * Actualizar () | | |
| **Observaciones:** | | |
| **Historia de Usuario** | | |
| **Número: 4** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de Estados | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador crear un nuevo estado para los préstamos en la aplicación.  Para crear un nuevo estado de préstamos se requiere como mínimo la siguiente información:   * id (auto numérico) * estado * descripción   **Funciones**   * Validaciones (). * Crear () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 5** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar Estados | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar un estado de la aplicación.  Para eliminar un estado se requiere como mínimo la siguiente información:   * Id.   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina el estado de ser estrictamente necesario de lo contrario se inhabilita. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 6** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de los estados. | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de un estado en la aplicación.  Para actualizar un estado se requiere como mínimo la siguiente información:   * id * estado * descripción   **Funciones**   * Validaciones (). * Actualizar () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 7** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de multas para los aportes | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador crear una nueva multa en la aplicación.  Para crear una nueva multa se requiere como mínimo la siguiente información:   * id (auto numérico) * nombre * descripción * precio * activo   **Funciones**   * Validaciones (). * Crear () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 8** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar multas de los aportes | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar una multa del sistema.  Para eliminar una multa se requiere como mínimo la siguiente información:   * Id.   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina la multa de ser estrictamente necesario de lo contrario se inhabilita. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 9** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de las multas. | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de una multa en el sistema.  Para actualizar un rol se requiere como mínimo la siguiente información:   * id * rol * descripción * activo   **Funciones**   * Validaciones (). * Actualizar () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 10** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de eventos | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador crear un nuevo evento en la aplicación.  Para crear un nuevo evento se requiere como mínimo la siguiente información:   * id (auto numérico) * fecha * ganancia(calculada) * responsable   **Funciones**   * Validaciones (). * Crear () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 11** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar eventos | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar un evento de la aplicación.  Para eliminar un rol se requiere como mínimo la siguiente información:   * Id.   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina un evento de ser estrictamente necesario ya que se puede perder información valiosa. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 12** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de los eventos. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de un evento en la aplicación.  Para actualizar un evento se requiere como mínimo la siguiente información:   * id * fecha * ganancia(calculada) * responsable   **Funciones**   * Validaciones (). * Actualizar () * Calcular ganncias del evento | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 13** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de aportes | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador registrar los aportes de un integrante en la aplicación.  Para registrar un aporte se requiere como mínimo la siguiente información:   * id (auto numérico) * integrante * evento * cuota * merienda * multa (si le impone una multa si quebranta alguna de las normas de la natillera) * mora (si posee días de atraso se le genera un interés de mora)   **Funciones**   * Validaciones (). * Crear () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 14** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar aportes | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar los aportes de un integrante de la aplicación.  Para eliminar un aporte se requiere como mínimo la siguiente información:   * Id.   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina un aporte de ser estrictamente necesario ya que puede perder información valiosa. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 1** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de los aportes. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de los aportes en la aplicación.  Para actualizar un rol se requiere como mínimo la siguiente información:   * id * integrante * evento * cuota * merienda * multa (si le impone una multa si quebranta alguna de las normas de la natillera) * mora (si posee días de atraso se le genera un interés de mora)   **Funciones**   * Validaciones (). * Actualizar () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 15** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de prestamos | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador registrar los préstamos que requieran los integrantes del sistema.  Para registrar un prestamo se requiere como mínimo la siguiente información:   * id (auto numérico) * usuario (persona que solicita el préstamo) * responsable (persona sirve como fiador) * monto prestado (cantidad que solicita) * saldo (es la totalidad de abonos que se han depositado) * estado (estado en que se encuentra actualmente el préstamo)   **Funciones**   * Validaciones (). * Crear () * Calcular el saldo * Calcular el cupo del préstamo | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 16** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar prestamos | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar los prestamos de un integrante de la aplicación, se realiza con el objetivo si algún usuario se le rechaza el préstamo.  Para eliminar un prestamos se requiere como mínimo la siguiente información:   * Id. * Estado (importante para saber que prestamos se deben eliminar)   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina un préstamo de ser estrictamente necesario ya que puede perder información valiosa. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 17** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de los préstamos. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de los préstamos en la aplicación, aplica cuando se quiera cambiar el estado del préstamo.  Para actualizar un rol se requiere como mínimo la siguiente información:   * id * usuario (persona que solicita el préstamo) * responsable (persona sirve como fiador) * monto prestado (cantidad que solicita) * saldo (es la totalidad de abonos que se han depositado) * estado (estado en que se encuentra actualmente el préstamo)   **Funciones**   * Validaciones (). * Actualizar () * Calcular saldo * Calcular cupo | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 18** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de abonos | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador registrar los abonos del prestamo del sistema.  Para registrar un abono se requiere como mínimo la siguiente información:   * id (auto numérico) * préstamo * interés (tasa de intereses se le está cobrando al abono) * cuota * fecha   **Funciones**   * Validaciones (). * Crear () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 19** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar abonos | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar los abonos de un préstamo de la aplicación.  Para eliminar un abono se requiere como mínimo la siguiente información:   * Id.   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina un préstamo de ser estrictamente necesario ya que puede perder información valiosa. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 20** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de los abonos. | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Alta |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de los abonos del préstamo en la aplicación.  Para actualizar un abono se requiere como mínimo la siguiente información:   * id (auto numérico) * préstamo * interés (tasa de intereses se le está cobrando al abono) * cuota * fecha   **Funciones**   * Validaciones (). * Actualizar () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 21** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir el registro de nuevos usuarios | | |
| **Prioridad en negocio:**  Alta | | **Riesgo en desarrollo:**  Media |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador crear un nuevo usuario en la aplicación.  Para crear un nuevo usuario se requiere como mínimo la siguiente información:   * id (auto numérico) * rol (relación con la tabla rol) * cedula * nombres * apellidos * email * teléfono * activo (por defecto true) * nick * password   **Funciones**   * Validaciones (). * Crear () | | |
| **Observaciones:**   * En una futura actualización se requiere añadir un campo avatar el cual sirva como imagen de perfil del usuario. * En una futura actualización se requiere validar la seguridad de la contraseña ingresada por el administrador. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 22** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir eliminar usuarios | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador eliminar un usuario de la aplicación.  Para eliminar un usuario se requiere como mínimo la siguiente información:   * Id.   **Funciones**   * Eliminar () | | |
| **Observaciones:**   * Solo se elimina el usuario de ser estrictamente necesario de lo contrario se inhabilita. * En una futura actualización se requiere enviar un mensaje al correo electrónico del usuario para informar que su usuario ha sido eliminado. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 23** | **Usuario:** Administrador. | |
| **Nombre historia:** Permitir actualizar la información de los usuarios. | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Administrador actualizar la información de un usuario en la aplicación.  Para actualizar un usuario se requiere como mínimo la siguiente información:   * id * rol (relación con la tabla rol) * cedula * nombres * apellidos * email * teléfono * activo * nick * password   **Funciones**   * Validaciones (). * Actualizar () | | |
| **Observaciones:**   * En una futura actualización se requiere añadir un campo avatar el cual sirva como imagen de perfil del usuario. * En una futura actualización se requiere validar la seguridad de la contraseña ingresada por el administrador. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 24** | **Usuario:** Integrante | |
| **Nombre historia:** Permitir visualizar los prestamos | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Integrante poder visualizar todos los préstamos que ha solicitado en el aplicativo.  Para visualizar el préstamo se requiere como mínimo la siguiente información:   * Id. * Responsable * Monto prestado * Estado * Saldo * Cupo libre   **Funciones**   * Visualizar () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 25** | **Usuario:** Integrante | |
| **Nombre historia:** Permitir visualizar los abonos | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Integrante poder visualizar todos los abonos que ha realizado a los respectivos prestamos en el aplicativo.  Para visualizar los abonos se requiere como mínimo la siguiente información:   * Préstamo * Interés * Cuota * fecha   **Funciones**   * Visualizar () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 26** | **Usuario:** Integrante | |
| **Nombre historia:** Permitir visualizar los aportes | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Integrante poder visualizar todos los aportes que ha realizado a la natillera.  Para visualizar los aportes se requiere como mínimo la siguiente información:   * evento * cuota * merienda * multa * mora   **Funciones**   * Visualizar () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 27** | **Usuario:** Integrante | |
| **Nombre historia:** Permitir registrarse | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Integrante poder registrarse como usuario del sistema natillera.  Para registrarse como usuario se requiere como mínimo la siguiente información:   * id * rol (por defecto es rol básico) * cedula * nombres * apellidos * email * teléfono * activo * nick * password   **Funciones**   * registrarse () * validaciones () | | |
| **Observaciones:** | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 28** | **Usuario:** Integrante | |
| **Nombre historia:** Permitir ingresar a la aplicacion | | |
| **Prioridad en negocio:**  Baja | | **Riesgo en desarrollo:**  Baja |
| **Puntos estimados:** | | **Iteración asignada:** |
| **Programador responsable:** Yording Cala | | |
| **Descripción:**  El sistema deberá ser capaz de permitir al Integrante ingresar a la aplicación.  Para registrarse como usuario se requiere como mínimo la siguiente información:   * email * nick * password * rol   **Funciones**   * autentificar () * validaciones () | | |
| **Observaciones:** | | |

Requisitos no Funcionales

* El sistema debe funcionar en tiempo real
* El sistema debe funcionar correctamente en cualquier navegador.
* El sistema debe tener 90% de disponibilidad semanalmente.
* El sistema debe correr en Android para facilitar en los eventos el registro de los aportes, préstamos y abonos.
* El sistema debe proporcionar al usuario una buena seguridad para no ser hackeado.

Restricciones

* Los sistemas operativos móviles no soportados será symbian y Windows pone
* La aplicación en Android será soportada por los dispositivos con Android 4.4 o superior.
* Los usuarios no podrán realizar transacciones bancarias con su dinero.
* La parte legal de la aplicación aún no se ha contemplado.
* La aplicación solo debe contener integrantes mayores de edad.

# ESPECIFICACIÓN ARQUITECTÓNICA

## CALIDADEDES SISTÉMICAS

A partir de los requisitos no funcionales describa las calidades sistémicas (puede tener como referencia el ISO25000) para describir las que son las más prioritarias para la arquitectura.

1. **Fiabilidad:** Capacidad de un sistema o componente para desempeñar las funciones especificadas, cuando se usa bajo unas condiciones y periodo de tiempo determinados. Esta característica se subdivide a su vez en las siguientes sub-características:

* Madurez. Capacidad del sistema para satisfacer las necesidades de fiabilidad en condiciones normales.
* Disponibilidad. Capacidad del sistema o componente de estar operativo y accesible para su uso cuando se requiere.
* Tolerancia a fallos. Capacidad del sistema o componente para operar según lo previsto en presencia de fallos hardware o software.
* Capacidad de recuperación. Capacidad del producto software para recuperar los datos directamente afectados y reestablecer el estado deseado del sistema en caso de interrupción o fallo.

1. **Seguridad:** Capacidad de protección de la información y los datos de manera que personas o sistemas no autorizados no puedan leerlos o modificarlos. Esta característica se subdivide a su vez en las siguientes sub-características:

* Confidencialidad. Capacidad de protección contra el acceso de datos e información no autorizados, ya sea accidental o deliberadamente.
* Integridad. Capacidad del sistema o componente para prevenir accesos o modificaciones no autorizados a datos o programas de ordenador.
* No repudio. Capacidad de demostrar las acciones o eventos que han tenido lugar, de manera que dichas acciones o eventos no puedan ser repudiados posteriormente.
* Responsabilidad. Capacidad de rastrear de forma inequívoca las acciones de una entidad.
* Autenticidad. Capacidad de demostrar la identidad de un sujeto o un recurso.

1. **Usabilidad:** Capacidad del producto software para ser entendido, aprendido, usado y resultar atractivo para el usuario, cuando se usa bajo determinadas condiciones. Esta característica se subdivide a su vez en las siguientes sub-características:

* Capacidad para reconocer su adecuación. Capacidad del producto que permite al usuario entender si el software es adecuado para sus necesidades.
* Capacidad de aprendizaje. Capacidad del producto que permite al usuario aprender su aplicación.
* Capacidad para ser usado. Capacidad del producto que permite al usuario operarlo y controlarlo con facilidad.
* Protección contra errores de usuario. Capacidad del sistema para proteger a los usuarios de hacer errores.
* Estética de la interfaz de usuario. Capacidad de la interfaz de usuario de agradar y satisfacer la interacción con el usuario.
* Accesibilidad. Capacidad del producto que permite que sea utilizado por usuarios con determinadas características y discapacidades.

1. **Portabilidad**: Capacidad del producto o componente de ser transferido de forma efectiva y eficiente de un entorno hardware, software, operacional o de utilización a otro. Esta característica se subdivide a su vez en las siguientes sub-características:

* Adaptabilidad. Capacidad del producto que le permite ser adaptado de forma efectiva y eficiente a diferentes entornos determinados de hardware, software, operacionales o de uso.
* Capacidad para ser instalado. Facilidad con la que el producto se puede instalar y/o desinstalar de forma exitosa en un determinado entorno.
* Capacidad para ser reemplazado. Capacidad del producto para ser utilizado en lugar de otro producto software determinado con el mismo propósito y en el mismo entorno.

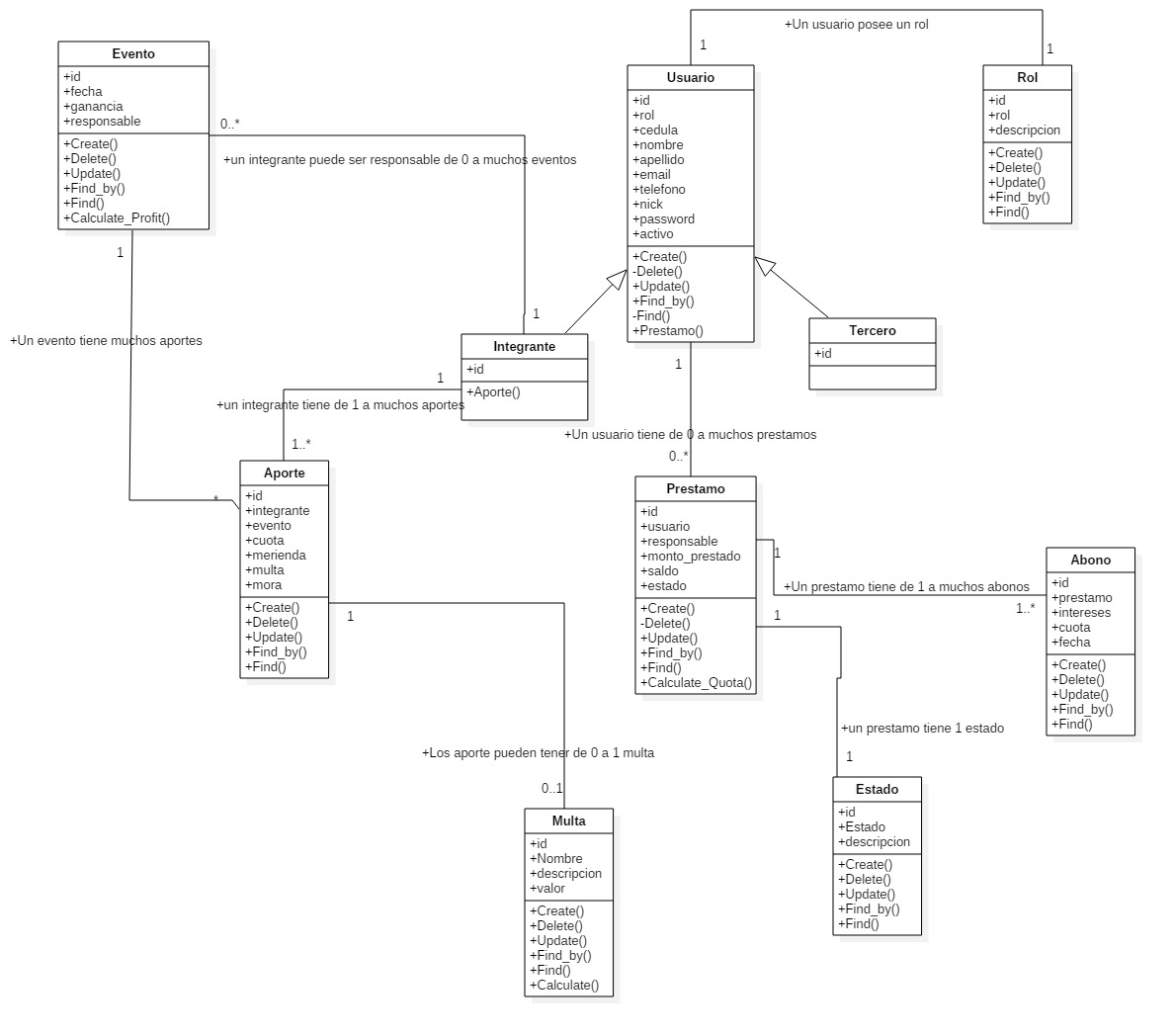
## ESTILO ARQUITECTÓNICO

El estilo arquitectónico del sistema está basado en un modelo cliente/servidor, donde se implementa una capa en la cual va alojado el front-end y se comunica por medio de protocolos resful a una capa externa donde se obtiene la información solicitada al backend.

## PATRONES Y FRAMEWORKS A UTILIZAR

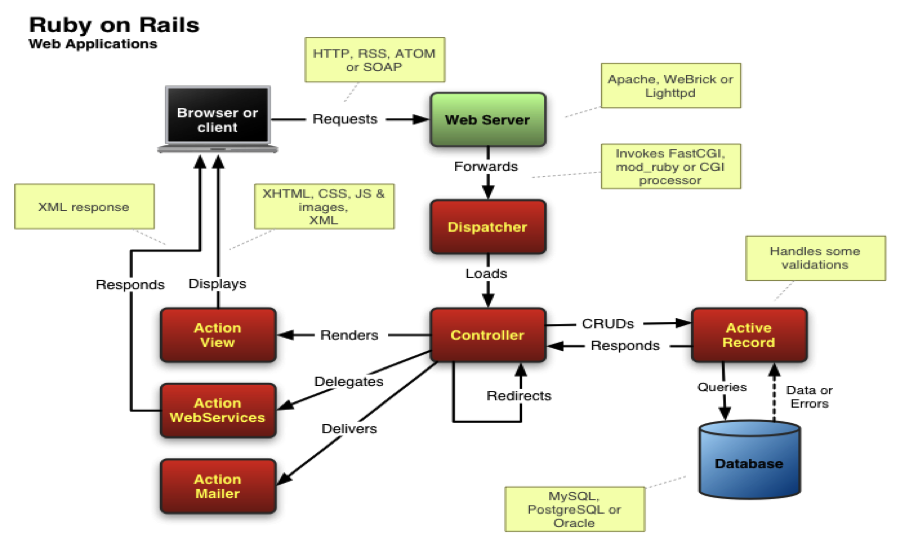
Patrones

* Patrón experto es utilizado para conocer la cantidad total pagada por el usuario en los abonos del préstamo.
* Patrón observer utilizado para que el cliente escuche los eventos que emite el servidor y poder así tener una aplicación de tiempo real.
* Patrón creador utilizado para crear nuevos objetos e instancias dentro de la aplicación, un ejemplo de esto es la clase Usuario encarga de crear nuevos objetos Integrante.
* Patrón controlador es utilizado para ser el intermediario entre la vista y un modelo, utilizado para intervenir en el crud de algunas entidades.



Frameworks

El frameworks en el cual se basa el sistema es ruby on rails el cual maneja una arquitectura MVC y que facilita las facilita mucho la generación de scaffolding completos para un determinado Modelo, además posee la mayoría de patrones utilizados para el desarrollo de la aplicación.



## IMPLEMENTACIÓN PARA LAS CALIDADES SISTËMICAS

Haga una matriz en donde por cada calidad sistémica, vs las capas de la aplicación de acuerdo al estilo arquitectónico, defina qué solución tecnológica usará, es decir que tecnología utilizará para garantizar el cumplimiento de esa calidad sistémica (En las filas las capas, en las columnas las calidades sistémicas, en la intercepción fila columna, la solución tecnológica)

## DISEÑO ARQUITÉCTONICA

Realice un diagrama en el que se puedan identificar los módulos arquitectónicos, las capas, los componentes de integración, protocolos y niveles que hacen parte de su arquitectura, puede usar un diagrama de componentes de UML si lo considera o un diagrama de bloque libre.

## VISTA DE DESPLIEGUE

Realice un diagrama final de cómo quedará distribuida la solución, puede utilizar un diagrama de despliegue de UML