**Documentación Proyecto**

**Plataforma de Mensajería Electrónica**

**NRC:** 2114

**Equipo No.** 2

**Integrantes:**

Lina Buelvas Lacombe

[lacombel@uninorte.edu.co](mailto:lacombel@uninorte.edu.co)

Jhon Pinedo Rojas

[jepinedo@uninorte.edu.co](mailto:jepinedo@uninorte.edu.co)

Christian Barrios Carrera

[cdcarrera@uninorte.edu.co](mailto:cdcarrera@uninorte.edu.co)

Jorge Ropero Gutierrez

jropero@uninorte.edu.co

**Descripción de roles del equipo**

|  |  |  |  |
| --- | --- | --- | --- |
| Rol | Integrante | Descripción | Tareas |
| SCRUM Master | Tutor | Persona encargada de llevar el seguimiento del proyecto. | Seguimiento del correcto desarrollo de las tareas del proyecto. |
| Desarrollador Front-End | ***Jorge Ropero Gutierrez*** | Persona encargada de realizar la parte visual de la aplicación. | Desarrollo de las vistas y estilos de la aplicación. |
| Desarrollador Back-End | **Christian Barrios Carrera** | Persona encargada de realizar la lógica de negocios de la aplicación. | Desarrollo de los controladores de la aplicación y diseño e integración de la base de datos. |
| Desarrollador Back-End | **Jhon Pinedo Rojas** | Persona encargada de realizar la lógica de negocios de la aplicación. | Desarrollo de los controladores de la aplicación y diseño e integración de la base de datos. |
| Desarrollador Full-Stack | **Jhon Pinedo Rojas** | Persona encargada de realizar la parte visual y la lógica de negocios de la aplicación. | Desarrollo de las vistas y estilos de la aplicación, de los controladores de la aplicación y diseño e integración de la base de datos. |
| Desarrollador Front-End | ***Lina Buelvas Lacombe*** | Persona encargada de realizar la parte visual de la aplicación. | Desarrollo de las vistas y estilos de la aplicación. |

**Definición de artefactos**

**Backlog Sprint 1**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación | Responsable |
| Definición de roles | Definición de los roles del equipo de trabajo del proyecto. | 1 horas | Todos los integrantes |
| Definición de artefactos | Definición de los artefactos de la metodología SCRUM para el desarrollo del proyecto. | 4 horas | Todos los integrantes |
| Diseño del diagrama de clases | Diseño del diagrama de clases de la aplicación a desarrollar. | 3 horas | Todos los integrantes |
| Definición del cronograma | Definición del cronograma de actividades del proyecto. | 3 horas | Todos los integrantes |

**Backlog Sprint 2**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación | Responsable |
| Definición del mapa de navegabilidad | Definición del mapa de navegabilidad que mostrará la distribución de las vistas de la aplicación | 4 horas | ***\*Jorge Ropero Gutierrez***  ***\** Christian Barrios Carrera** |
| Selección de la plantilla de estilos | Selección de la librería CSS a usar para los estilos de las vistas de la aplicación. | 3 hora | ***Lina Buelvas Lacombe*** |
| Diseño e implementación de las vistas | Diseño e implementación de las vistas en HTML y CSS. | 30 horas | **\*Jhon Pinedo Rojas**  ***\*Lina Buelvas Lacombe***  **\*Christian Barrios Carrera**  **\* *Jorge Ropero Gutierrez*** |
| Creación del proyecto en GIT | Creación del proyecto en GIT y posterior cargue a la nube en la plataforma Github. | 1 hora | ***Lina Buelvas Lacombe*** |

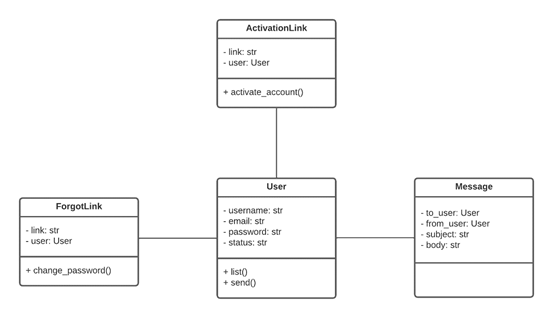
**Backlog Sprint 3**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación | Responsable |
| Diseño e implementación de los controladores para formularios y otras funcionalidades | Diseño e implementación de los controladores que representan la lógica de negocios de la aplicación. | 50 horas | **\*Jhon Pinedo Rojas**  **\*Christian Barrios Carrera** |
| Diseño e implementación de base de datos | Diseño de la base de datos relacional que alojará los datos de la aplicación e implementación de la misma usando el motor SQLite. | 5 horas | **\*Jhon Pinedo Rojas**  ***\*Jorge Ropero Gutierrez*** |
| Desarrollo de integración de controladores y bases de datos | Integración de la base de datos con los controladores para la búsqueda y almacenamiento de información persistente de manera segura. | 20 horas | **\*Jhon Pinedo Rojas**  ***\*Jorge Ropero Gutierrez*** |
| Diseño e implementación de portal de acceso usando método de autenticación basado en usuario y contraseña | Diseño e implementación del método de autenticación de los usuarios y manejo de sesiones. | 15 horas | **\*Jhon Pinedo Rojas**  ***\*Lina Buelvas Lacombe*** |

**Backlog Sprint 4**

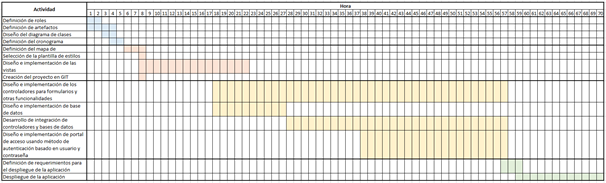
|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación | Responsable |
| Definición de requerimientos para el despliegue de la aplicación | Definición de los requerimientos necesarios para realizar el despliegue de la aplicación en la plataforma PythonAnywhere | 3 horas | Todos los integrantes |
| Despliegue de la aplicación | Configuración, despliegue y verificación del funcionamiento de la aplicación en la plataforma PythonAnywhere | 12 horas | Todos los integrantes |

**Diagrama de clases**

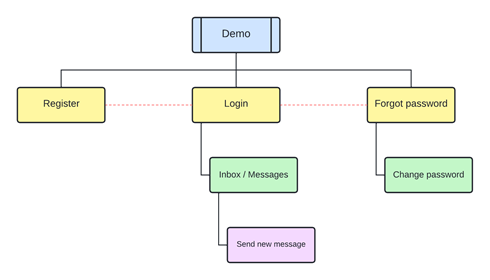


La clase usuario tendrá la información referente al usuario, es decir, su nombre de usuario, su correo, su contraseña y el estado de su cuenta. La clase mensaje tendrá la información de los mensajes de la plataforma, es decir, el usuario que envía el mensaje, el usuario que lo recibe, el asunto y el cuerpo del mensaje.

**Cronograma de tareas**



**Mapa de navegabilidad**



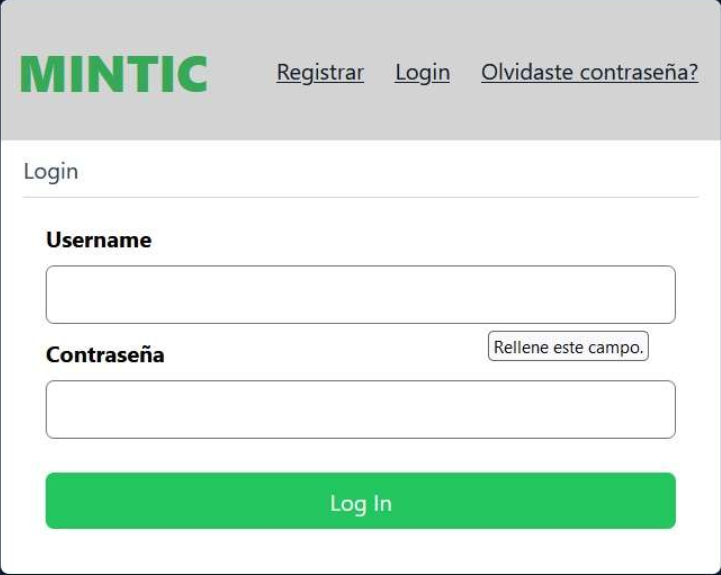
La aplicación en su pantalla principal tiene 3 opciones, estas son el registro, el inicio de sesión y la recuperación de contraseña, y se puede desplazar entre ellas. La pantalla de recuperación de contraseña, dará paso a la pantalla de cambio de contraseña al acceder a ella mediante el enlace enviado. La pantalla de inicio de sesión da paso a la de visualización de mensajes y esta a su vez da paso a la pantalla de envío de nuevo mensaje.

**Vistas de la aplicación**

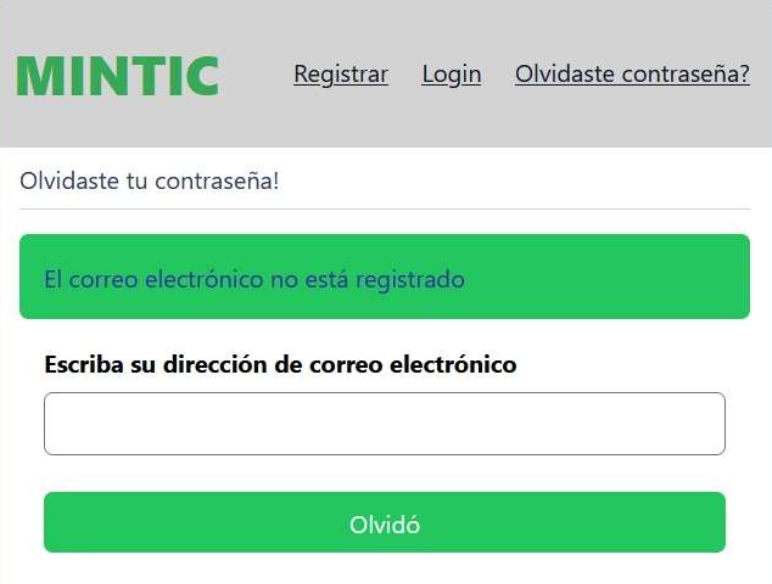
1. **Vista de registro:**



**2. Vista de inicio de sesión:**



1. **Vista de solicitud de recuperación de contraseña:**



1. **Vista de cambio/recuperación de contraseña:**



1. **Vista de visualización de mensajes:**



1. **Vista de envío de un nuevo mensaje:**

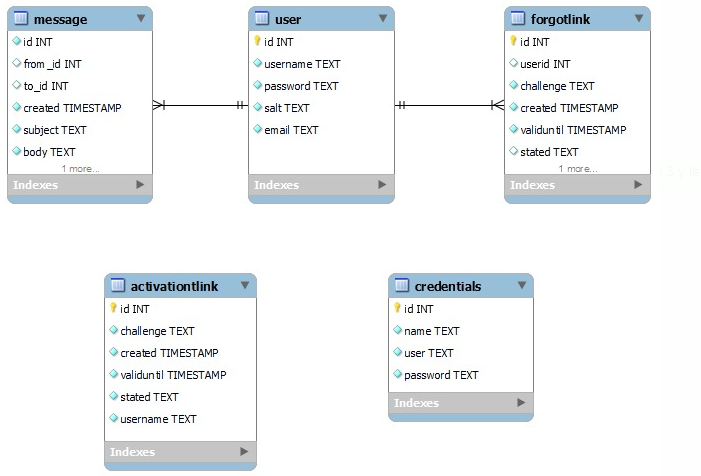
Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Métodos controladores del Back-End** (*Sprint 3*)

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre | Ruta | Métodos HTTP permitidos | Lógica algorítmica |
| Register | **/register** | **POST** | Con este método validamos que los datos ingresados por los usuarios, cumplan con los criterios definidos para las variables de entrada, con lo cual podemos verificar que los campos no estén en blanco, que el usuario no se encuentre ya registrado en la base de datos, además inserta la contraseña encriptada y el resto de datos a la tabla “activationlink”. Por último, envía enlace de activación a la cuenta de correo registrada. |
| Activate | **/activate** | **GET** | Con esta función validamos que el enlace de activación enviado al correo del usuario, este vigente y que este coincida con el enviado, luego actualiza el estado del campo state de la tabla “activationlink” e inserta los datos de registro que el usuario ingreso, en la tabla “user” posterior a esto redirige al template login.html |
| Login | **/login** | **POST** | Con este método se valida:   * Si existe una sesión activa se redirige al inbox.show * Que los campos de en el recuadro de login no estén vacíos. * Que los datos ingresados coincidan con los registrados en la base de datos. * Redirige al template show.html, si se cumplen las validaciones, sino permanece en el template login.html |
| Forgot | **/forgot** | **POST** | Este método realiza las siguientes validaciones:   * Si existe una sesión activa redirige al inbox.show * Valida que el campo correo no se encuentre en blanco y que este registrado en la base de datos * Actualiza el campo state de la tabla forgotlink * Envía enlace de cambio de contraseña al correo registrado. |
| Change | **/Change** | **GET** | Con esta función se valida:   * Si hay sesión activa redirige al inbox.show * Valida que el enlace de cambio de contraseña enviado al correo del solicitante, todavía este vigente y que este coincida con el enviado. * El enlace redirige al template change.html |
| Confirm | **/confirm** | **POST** | Esta función realiza la siguiente rutina.   * Si hay sesión activa redirige al inbox.show * Valida que los campos de cambio de contraseñas no estén vacíos y que además ambos coincidan. Sino renderiza el template change.html. * Si se cumplen las validaciones, actualiza el campo state de la tabla “forgotlink” * Actualiza el campo password de la tabla “user”. * Y Redirige al template de login.html. |
| Logout | **/logout** |  | Este método cierra la sesión y redirige al template login.html |
| Show | **/show** |  | Esta función renderiza el template show.html donde aparecen los mensajes que han enviado a mi usuario. |
| Send | **/send** | **POST** | Esta función ejecuta la siguiente lógica algorítmica.   * Valida que los campos to\_id, subject, body no estén vacíos. * Valida que el destinatario exista en la base de datos. * Guarda en la tabla “message” la información correspondiente al mensaje enviado. * Si cumple con todas las validaciones redirige al template show.html, sino renderiza el template send.html |

**Base de datos** (*Sprint 3*)



En el diagrama se representan cinco tablas:

* La tabla “user” será la encargada de almacenar la información de los usuarios. Esta tiene una relación de uno a muchos con la tabla “message” y con la tabla “forgotlink”.
* La tabla “activationtlink” tendrá la información sobre la generación de enlaces de activación.
* La tabla “credentials” tendrá los datos de la cuenta que genera los enlaces de activación y recuperación de contraseña.

Como se observa las dos últimas tablas no cuenta con relaciones entre sí, ni con las otras tablas

**Prácticas de programación segura** (*Sprint 3*)

Descripción de las buenas prácticas de programación segura usadas en ña aplicación:

* **Uso de librerías seguras:** En el proyecto se utilizan librerías seguras para generar y validar passwords encriptados (from werkzeug.security import check\_password\_hash, generate\_password\_hash). Además, uso de librería segura para Manejo de sesión.
* **Validación de entrada:** Como una buena práctica en el diseño de la aplicación, mediante el diseño del frontend se establece la validación de los datos de entrada para evitar errores antes de que sean procesados.
* **Estandarización**: El desarrollo se realizó siguiendo la plantilla estándar proporcionada, lo que le permite a cualquier miembro del equipo y al tutor encargado identificar más fácilmente las diferentes líneas de código.
* **Evitar vulnerabilidades:** Para evitar vulnerabilidades de acceso y modificación de los datos, no se ejecuta sentencias QUERY concatenando las variables para prevenir el ataque de inyección SQL a la base de datos*.*

***ENLACE DE REPOSITORIO.***

<https://github.com/jhonpinrojas/Spring3ciclo3>