**Documentación Proyecto**

**Plataforma de Gestión de Notas**

**NRC:** 2093

**Equipo No.** 05

**Integrantes:**

*Alejandro Vargas Castillo*

*Dana Lorena Montes Gamboa*

*Lida Yohaira Hernández Nieto*

*Jc Biigvai Sierra Ojeda*

*Nahomi Cuello Garcia*

**Descripción de roles del equipo** (*Sprint 1*)

|  |  |  |  |
| --- | --- | --- | --- |
| Rol | Integrante | Descripción | Tareas |
| SCRUM  Master | **Yurley Katherine Echeverria Leal** | **Persona encargada de llevar el seguimiento del proyecto.** | **Seguimiento del correcto desarrollo de las tareas del proyecto.** |
| Team Lead | **Alejandro Vargas Castillo** | **Encargado de realizar seguimiento al proyecto con el objetivo de que se cumplan los requerimientos en términos de funcionalidad y tiempos** | 1. **Verificar funcionamiento y desarrollo del aplicativo.** 2. **Realizar seguimiento a cada uno de los miembros del equipo sobre el cumplimiento de sus funciones.** 3. **Apoyar los procesos en los cuales los miembros del equipo requieran apoyo.** 4. **Entregar cada uno de los entregables por Sprint.** 5. **Organizar los documentos** |
| Quallity assurance - Fullstack | **Lida Yohaira Hernández Nieto** | **Encargada de implementar partes visuales y partes lógicas en el proyecto con el fin de asegurar que el producto cumpla con los el mínimo producto viable** | 1. **Programar verificar correcto funcionamiento del producto final.** 2. **Apoyar la programación del back-end y front-end.** 3. **Implementación del HTML y CSS** |
| Desarrollador Backend | **Jc Biigvai Sierra Ojeda** | **Se encarga de de desarrollar la capa lógica de programación que hace que una página web funcione y se conecte correctamente con la base de datos.** | 1. **Programar, verificar funcionamiento sistema backend.** 2. **Implementación de los controladores** 3. **Implementación de la ruta de la página.** 4. **Diseño de los modelos** |
| Desarrollador  FrontEnd | **Dana Lorena Montes Gamboa** | **Encargado del diseño y desarrollo web de la interfaz del producto, así como de crear la interactividad con los usuarios.** | 1. **Programar y verificar funcionamiento código frontend** 2. **Implementación del html.** 3. **Implementación del css.** 4. **Implementación del javascript.** |
| Base de datos | **Nahomi Cuello Garcia** | Encargado de Administrar, procesar y almacenar los datos para que puedan ser usados de forma accesible y fiable. | 1. **Creación de la relación de base de datos** 2. **Desarrollar, testear y hacer mantenimiento de las bases de datos.** |

**Definición de artefactos** (*Sprint 1*)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PRODUCT BACKLOG** | | | | |
| Nombre de la tarea | Prioridad | Estado | Puntos de historia | Asignado al Sprint |
| **Sprint 1** | | | | |
| Asignación de roles | MEDIA | FINALIZADO | 1 | SI |
| Definición de artefactos | BAJA | FINALIZADO | 2 | SI |
| Diagrama de clases | ALTA | FINALIZADO | 4 | SI |
| Cronograma de tareas | MEDIA | FINALIZADO | 1 | SI |
| **Sprint 2** | | | | |
| Mapa de navegabilidad | BAJA | FINALIZADO | 2 | SI |
| Interfaz gráfica de la aplicación | MEDIA | FINALIZADO | 2 | SI |
| Seleccionar estilos de vista | MEDIA | FINALIZADO | 2 | SI |
| Proyecto GIT | ALTA | FINALIZADO | 3 | SI |
| **Sprint 3** | | | | |
| Diseñar controladores para formularios | ALTA | FINALIZADO | 3 | SI |
| Diseñar base de datos | ALTA | FINALIZADO | 4 | SI |
| Integración de controladores | MEDIA | FINALIZADO | 4 | SI |
| Portal de acceso | MEDIA | FINALIZADO | 4 | SI |
| **Sprint 4** | | | | |
| Requerimientos para la aplicación | MEDIA | SIN EMPEZAR | 4 | SI |
| Configuración de la aplicación | MEDIA | SIN EMPEZAR | 4 | SI |
| Funcionamiento de la aplicación | MEDIA | SIN EMPEZAR | 4 | SI |

**Backlog Sprint 1**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación (Horas) | Responsable |
| Asignación de roles | Establecer los roles a trabajar, las tareas y la descripción del trabajo que realizara cada miembro del equipo. | 2h | Dana Montes Jc Sierra |
| Definición de artefactos | Realizar listas de tareas que se va a trabajar en cada sprint y otra lista, para las tareas a realizar durante todo el proyecto. | 8h | Dana Montes  Jc Sierra |
| Diagrama de clases | Diagramar UML y proponer las clases para trabajar el proyecto. | 8h | Alejandro Vargas |
| Cronograma de tareas | Asignar fechas para las actividades definidas para cada sprint. | 4h | Jc Sierra Nahomi Cuello |

**Backlog Sprint 2**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación (Horas) | Responsable |
| Mapa de navegabilidad | Diseñar mapa de la aplicación y organizar los contenidos a visualizar. | 6h | Dana Montes |
| Interfaz gráfica de la aplicación | Diseñar bocetos de las vistas de la aplicación | 8h | Alejandro Vargas |
| Seleccionar estilos de vista | Implementar los estilos de vista | 6h | Lida Hernández  Nahomi Cuello |
| Proyecto GIT | Crear proyecto en la plataforma | 10h | Jc Sierra |

**Backlog Sprint 3**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación (Horas) | Responsable |
| Diseñar controladores para formularios | Definir ruta, métodos HTTP permitidos y diseño a implementar | 12h | Jc Sierra  Dana Montes |
| Diseñar base de datos | Diseñar diagrama relacional, implementar tablas de bases de datos con SQLite y definir las relaciones más importantes. | 14h | Nahomi Cuello |
| Integración de controladores | Validar entrada de datos, usar librerías, actualizar bases de datos. | 8h | Lida Hernández |
| Portal de acceso | Diseñar y crear sesiones utilizando funciones hash criptográficas para almacenar contraseñas usando salts. | 8h | Alejandro Vargas |

**Backlog Sprint 4**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación (Horas) | Responsable |
| Requerimientos para la aplicación | Realizar un documento descriptivo de los requerimientos y el proceso de despliegue para la aplicación | 8h | Jc Sierra  Alejandro Vargas |
| Configuración de la aplicación | Utilizar la plataforma (PaaS) para la configuración y despliegue de la aplicación(Heroku) | 6h | Nahomi Cuello  Dana Montes |
| Funcionamiento de la aplicación | Verificar correcto funcionamiento de la aplicación y conectividad al dominio web. | 3h | Lida Hernández |

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Archivo** | **Ruta** | **Métodos HTTP permitidos** | **Lógica algorítmica** |
| **auth.py** | **@bp.route(‘/activate’)** | **GET, POST** |  |
| **auth.py** | **@bp.route(‘/register’)** | **GET, POST** |  |
| **auth.py** | **@bp.route(‘/confirm’)** | **GET, POST** |  |
| **auth.py** | **@bp.route(‘/change’)** | **GET, POST** |  |
| **auth.py** | **@bp.route(‘/forgot’)** | **GET, POST** |  |
| **auth.py** | **@bp.route(‘/login’)** | **GET, POST** |  |
| **auth.py** | **@bp.route(‘/logout’)** | **GET, POST** |  |
| **\_\_init\_\_.py** | **@bp.route(‘/send’)** | **GET, POST** |  |
| **\_\_init\_\_.py** | **@bp.route(‘/show’)** | **GET, POST** |  |

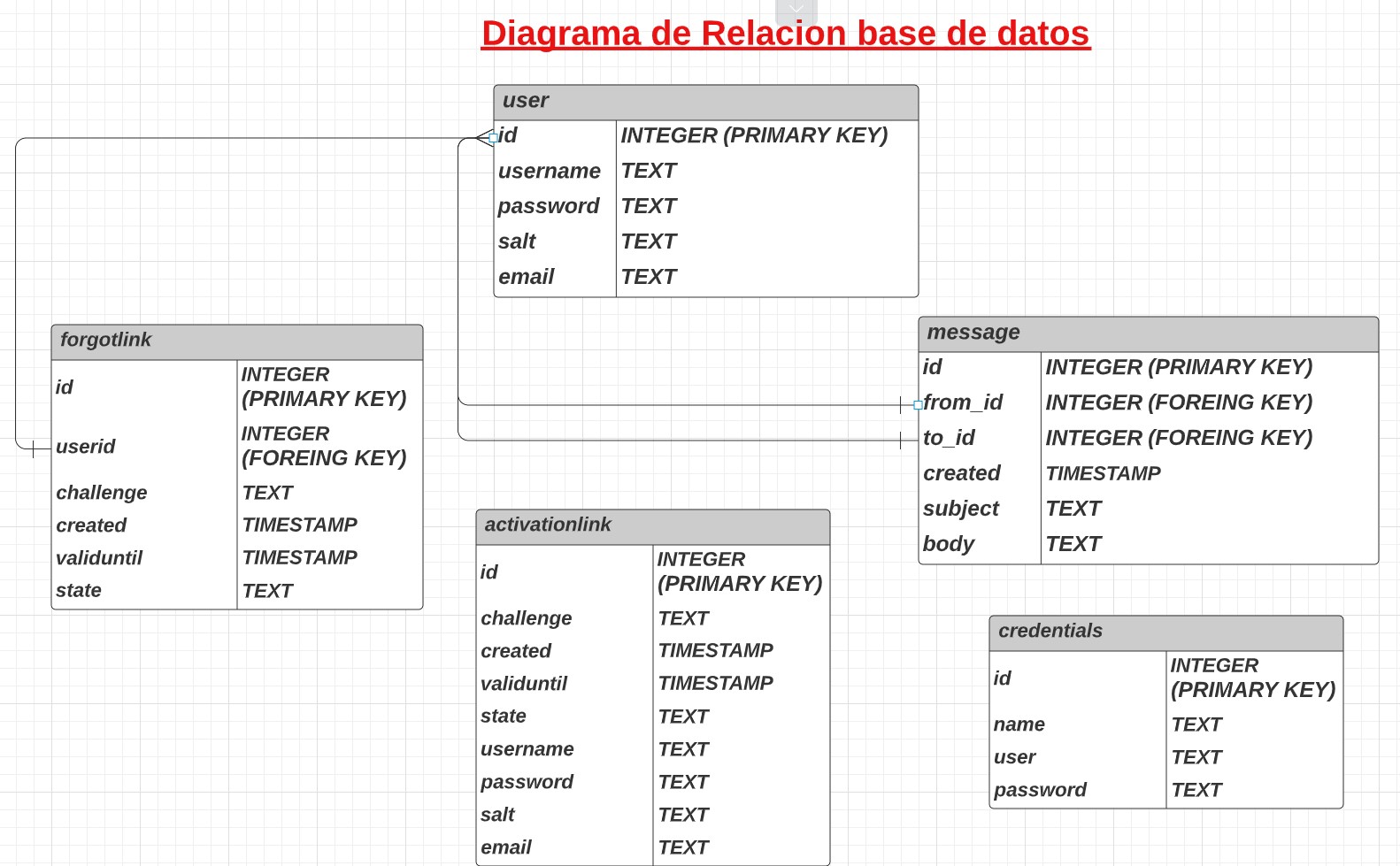
1. Nombre método Register
   * Ruta: @bp.route(‘/register’)
   * Métodos HTTP permitidos: GET, POST
   * **Lógica algorítmica:** este método recibe los datos iniciales para la creación en la base de datos, se solicita al usuario escribir su username, crear una contraseña, digitar un email que sea correo electrónico. Realiza todas las validaciones necesarias para el funcionamiento correcto y le da al botón registrar, si cumple con todos los requisitos redirecciona a (‘/activate’) y espera un funcionamiento del controlador activate. Si hay algún error se despliega flash error.

1. Nombre método controlador Activate
   * Ruta: @bp.route('/activate')
   * Métodos HTTP permitidos: GET, POST
   * Lógica algorítmica: este método no puede funcionar activar sin haber hecho el registro ya que el usuario necesita usar el link que es enviado a su correo electrónico y dar clic en este para pasar la activación de la cuenta
2. Nombre método controlador Confirm
   * Ruta: @bp.route(‘/confirm’)
   * Métodos HTTP permitidos: **GET, POST**
   * Lógica algorítmica: Este método sirve cuando el usuario tiene que digitar su nueva contraseña dos veces para confirmar que digita correctamente su nueva contraseña.
3. Nombre método controlador Change
   * Ruta: @bp.route(‘/change’)
   * Métodos HTTP permitidos: GET**,POST**
   * **Lógica algorítmica:** Este método sirve cuando un usuario existente en la base de datos, olvida su contraseña y viene de usar el controlador forgot que generar un link para cambiar su contraseña. Se verifica en la base de datos que sea un usuario activo y ejecuta el cambio de contraseña.
4. Nombre método controlador Forgot
   * Ruta: @bp.route(‘/forgot’)
   * Métodos HTTP permitidos: **GET, POST**
   * **Lógica algorítmica:** cuando se presiona en olvidar contraseña se activa este método, se solicita el ingreso del correo registrado y se verifica que sea válido si no lo es arroja error, si coinciden las credenciales se envía el correo de autorización de nueva contraseña y procede a cambiar la contraseña, lo redirecciona a ('/change')activando el método change.
5. Nombre método controlador Login
   * Ruta: @bp.route(‘/login’)
   * Métodos HTTP permitidos: **GET, POST**
   * **Lógica algorítmica:** lee los datos del usuario, id y contraseña, primero verifica que los campos estén completos o si no devuelve error. Si los campos se diligenciaron prosigue a verificar que los datos coincidan con los datos almacenados en la base de datos, si encuentra un error devuelve flash y sigue en la misma página dando instrucciones al usuario para que pueda hacerlo correcto.

Si todo se ejecuta correctamente te lleva a la siguiente página para escribir el mensaje (/show).

1. Nombre método controlador Logout
   * Ruta: @bp.route(‘/logout’)
   * Métodos HTTP permitidos: GET, POST
   * Lógica algorítmica: es el método que permite cerrar sesión y redirige a la página de inicio de sesión.
2. Nombre método controlador Send
   * Ruta: @bp.route('/send')
   * Métodos HTTP permitidos: GET, POST
   * Lógica algorítmica: requiere de inicio de sesión previo, para ejecutar el método y envíe el mensaje correctamente, debe llenar todos los campos, son obligatorios username, body and subject. Si no cumple, devuelve alerta de error, debe llenar el campo. Si todo se llena correctamente, se envía el mensaje y se almacena en la base de datos.
3. Nombre método controlador Show
   * Ruta: @bp.route(‘/show’)
   * Métodos HTTP permitidos: GET, POST
   * Lógica algorítmica: es el método que despliega la bandeja de entrada, recibe la información de la base de datos, se ejecuta y devuelve los mensajes almacenados.

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



*La base de datos tiene 5 tablas para la función de la aplicación*

***User:*** *en esta se guarda la información del usuario: correo, username, contraseña y hash para verificar que este usuario es el que se registró en la base de datos.*

***Message:*** *la tabla en la que se guardan los mensajes que se envían los usuarios de la aplicación aquí se guarda el id del mensaje el id del usuario que lo envía tanto como el que lo recibe la fecha de creación, el asunto y el contenido del mensaje.*

***Credentials:*** *aquí se implementan las credenciales del correo electrónico que se usa para enviar los links de activación de cuenta y olvido de contraseña.*

***Activationlink:*** *la tabla para la activación de cuenta, aquí se guardarán los datos necesarios para el envío del link de activación al correo ingresado, tiempo en que el link será válido para usarse y el salt para la generación del código de confirmación único.*

***Forgotlink:*** *esta tabla es cual el usuario olvide su contraseña se le enviara un link único se guardará y también tendrá una duración de validez para que genere una nueva contraseña.*

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

*Para nuestra aplicación implementamos diferentes librerías, y funciones hash para ejecutar los procesos y almacenar datos de manera segura.*

*MarkupSafe==2.0.1*

*Con este se mitigan los ataques de inyección, lo que significa que la entrada de un usuario que no es de confianza se puede mostrar de forma segura en una página.*

*De la librería werkzeug.security utilizamos check\_password\_hash y generate\_password\_hash*

*itsdangerous==2.0.1.*

*hashP = generate\_password\_hash(password + salt)*

*Lo implementamos para generar una clave secreta, en nuestra app implementamos salt combinado con os.urandom() para enviar el email de activación de la cuenta.*

*asegurando la seguridad utilizamos esta librería y así usaríamos diferentes códigos aleatorios para cada usuario.*