**Documentación Proyecto**

**Plataforma de Mensajería Electrónica**

**NRC:** 2113

**Equipo No.** 12

**Integrantes:**

*JUAN PABLO MORENO (100%)*

*VICTOR ALFONSO PATIÑO (5%)*

*JUAN SEBASTIAN NOVOA (5%)*

*ANGELA MELISSA MERCADO (20%)*

*JULIANA PAOLA MARTINEZ (5%)*

**Descripción de roles del equipo**

|  |  |  |  |
| --- | --- | --- | --- |
| Rol | Integrante | Descripción | Tareas |
| SCRUM Master | (Tutor)  **Kevin Santiago Tocora** | Persona encargada de llevar el seguimiento del proyecto. | Seguimiento del correcto desarrollo de las tareas del proyecto. |
| Líder del equipo | **Juan Pablo Moreno** | Persona encargada de coordinar las entregas y que se cumplan con los entregables de cada sprint | Mantener informados a los integrantes del equipo de las indicaciones del tutor.  Revisar que las entregas cumplan con los objetivos del enunciado.  Apoyar a los desarrolladores en tareas específicas de horas de ruta y codificación. |
| Desarrollador Front-End | **Angela Melissa Mercado** | Persona encargada de realizar la parte visual de la aplicación. | Desarrollo de las vistas y estilos de la aplicación. |
| Desarrollador Back-End | **Juan Sebastián Novoa** | 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 | **Juliana Paola Martínez** | 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 | **Víctor Alfonso Patiño** | 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. |

**Definición de artefactos**

**Backlog Sprint 1**

|  |  |  |  |
| --- | --- | --- | --- |
| User story | Descripción | Estimación | Responsable |
| Definición de roles | Definición de los roles de los integrantes del equipo de trabajo del proyecto. | 2 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. | 2 horas | Todos los integrantes |
| Definición del cronograma | Definición del cronograma de actividades del proyecto. | 2 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 | 3 horas | Integrante 1 y 4 |
| Selección de la plantilla de estilos | Selección de la librería CSS a usar para los estilos de las vistas de la aplicación. | 1 hora | Integrante 1 y 4 |
| Diseño e implementación de las vistas | Diseño e implementación de las vistas en HTML y CSS. | 15 horas | Integrante 1 y 4 |
| Creación del proyecto en GIT | Creación del proyecto en GIT y posterior cargue a la nube en la plataforma Github. | 1 hora | Integrante 1 |

**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. | 40 horas | Integrante 2 y 3 |
| 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. | 10 horas | Integrante 4 |
| 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. | 30 horas | Integrante 2 y 3 |
| 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. | 20 horas | Integrante 4 |

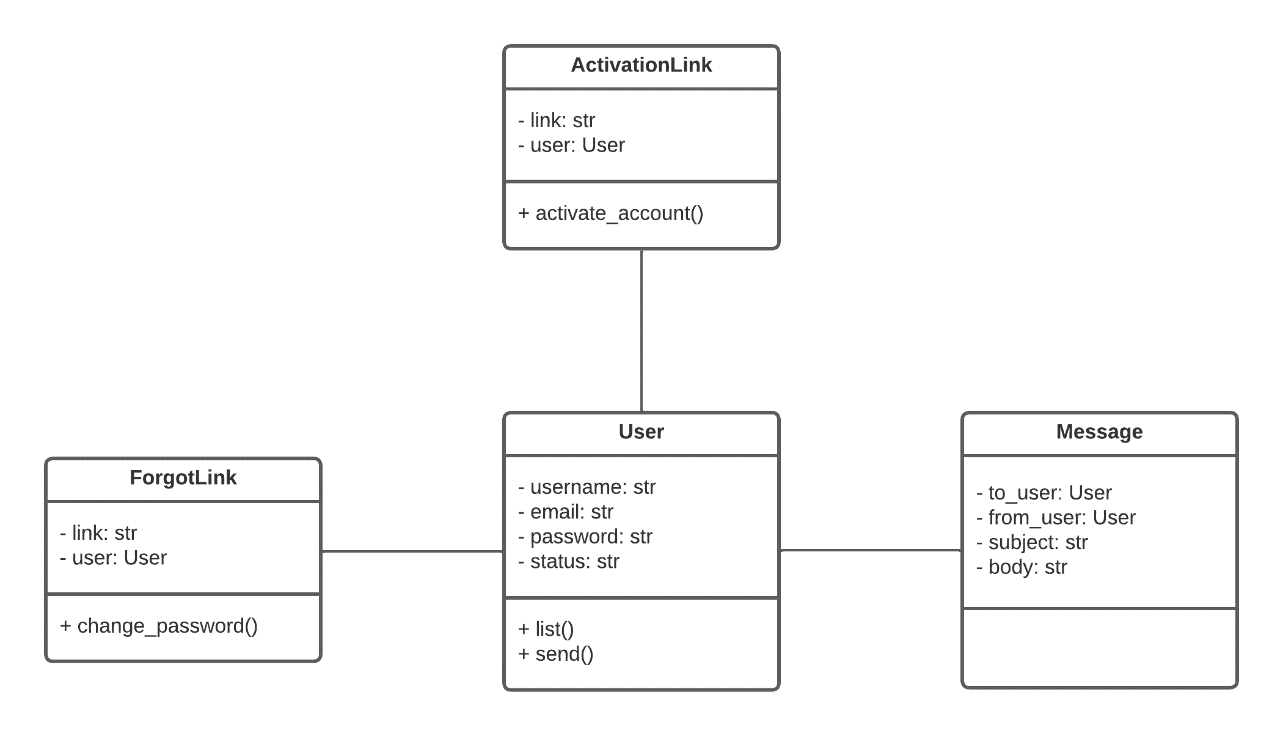
**Backlog Sprint 4**

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

**Link repositorio del Sprint 3** (*Sprint 3*)

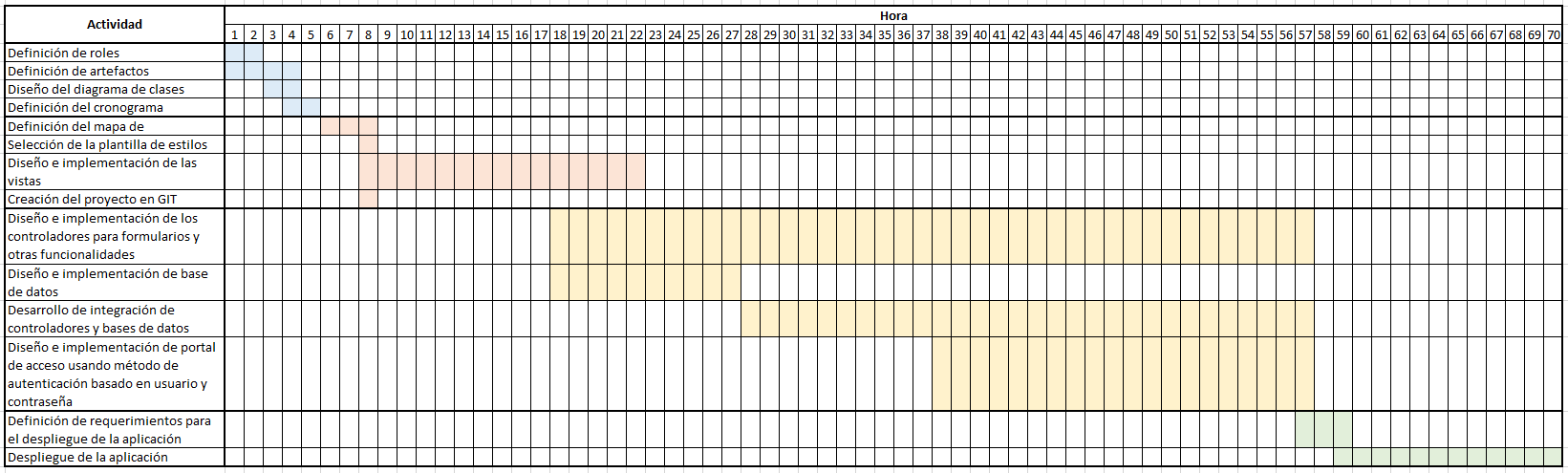
<https://github.com/jpml1771/Grupo12_2113/tree/main/Sprint_3>

**Diagrama de clases**

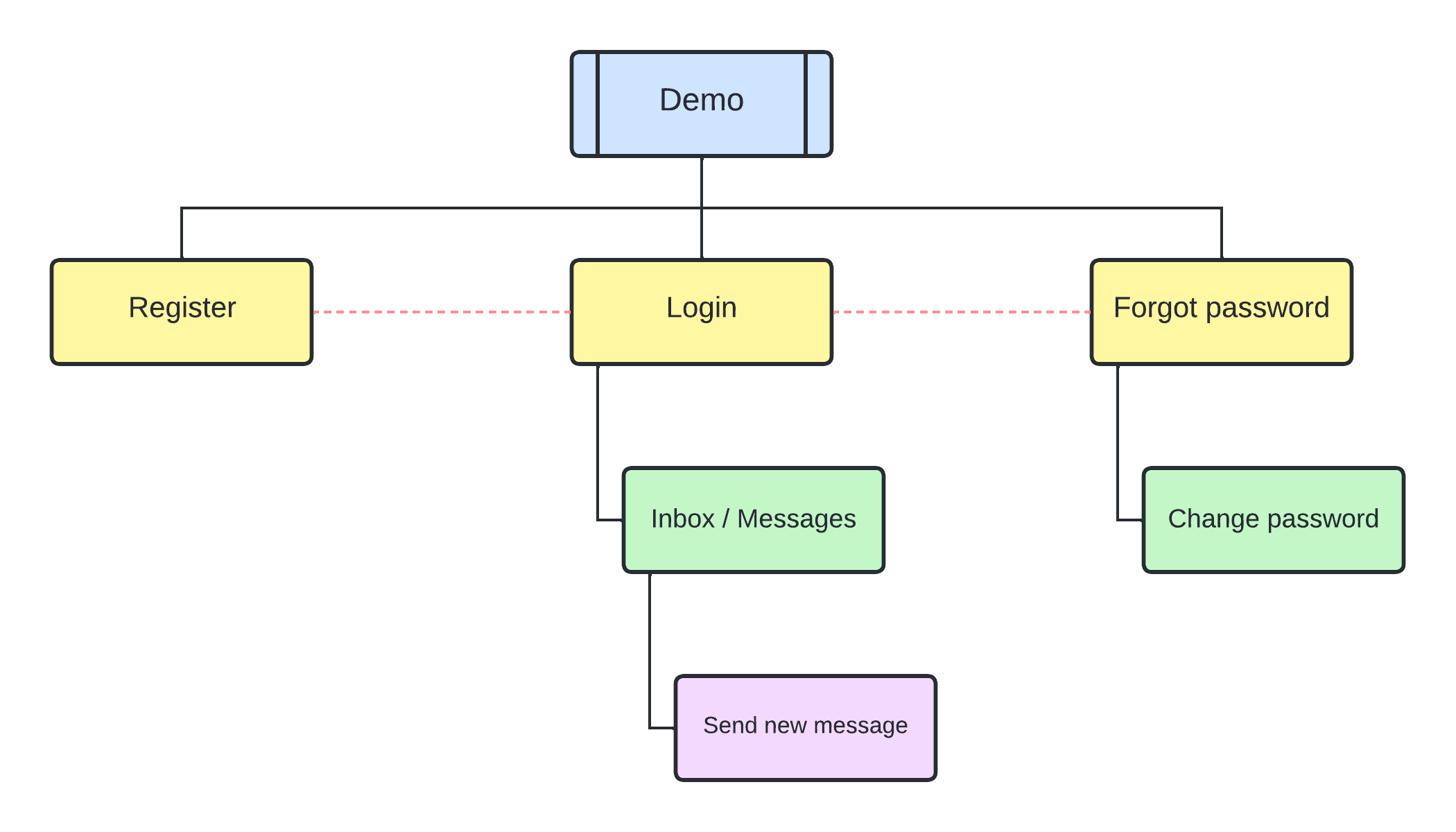


La clase usuario tendrá la información referente al usuario, es decir, su nombre de usuario, su correo, su contraseña y el estado de su cuenta. La clase mensaje tendrá la información de los mensajes de la plataforma, es decir, el usuario que envía el mensaje, el usuario que lo recibe, el asunto y el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

* **Vista de registro:**

Graphical user interface, application

Description automatically generated

* **Vista de inicio de sesión:**

**Graphical user interface, application, Teams

Description automatically generated**

* **Vista de solicitud de recuperación de contraseña:**

Graphical user interface, application, Teams

Description automatically generated

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

Graphical user interface, application

Description automatically generated

* **Vista de visualización de mensajes:**

Graphical user interface, application

Description automatically generated

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

**Graphical user interface, application, Teams

Description automatically generated**

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

* Nombre método activate()
  + Ruta: /activate
  + Métodos HTTP permitidos: ('GET')
  + Lógica algorítmica:

*El controlador permite establecer si el usuario está activado en la sesión para mostrar el inbox del user autenticado. El método verifica si el link de activación tiene un número challenge y si el usuario no está confirmado. Si es correcto el método ingresa el usuario a la base de datos por medio de los datos temporales de activationlink. Si hay fallos redirige a página de login.*

* Nombre método register()
  + Ruta: /register
  + Métodos HTTP permitidos: ('POST')
  + Lógica algorítmica:

*El método permite registrar un usuario nuevo a partir de la información del formulario de la página de register.html, el método también permite verificar si los datos ingresados son correctos o se envían mensajes flash error y redirigiendo nuevamente a la pagina de registro. Al estar correctos los datos, se invoca el método de envío de email de activación al correo y se transfiere a la página de login.html.*

* Nombre método confirm()
  + Ruta: /confirm
  + Métodos HTTP permitidos: ('POST')
  + Lógica algorítmica:

*El método permite al usuario cambiar la contraseña después que ha invocado los métodos change e ingresado al link de cambio de contraseña. Por medio de la verificación del id asociado a la tabla user, se verifica que exista un número de cambio de contraseña y estado activo, si es correcto este método permite dirigirse a la pagina change.html e ingresar una nueva contraseña. Esta contraseña se ingresa dos veces para verificación. Después de esta confirmación se cambia el estado del registro forgotlink a inactivo según el id del usuario y se actualiza la contraseña en la tabla user con encriptada por medio de un hash y una semilla salt.*

* Nombre método change ()
  + Ruta: / change
  + Métodos HTTP permitidos: ('GET')
  + Lógica algorítmica:

*El método confirma si existe un registro activo de forgotlink para el usuario según el id de solicitud, si es correcto se remite a la pagína change.html. De lo contrario se remite nuevamente a la página forgot.html.*

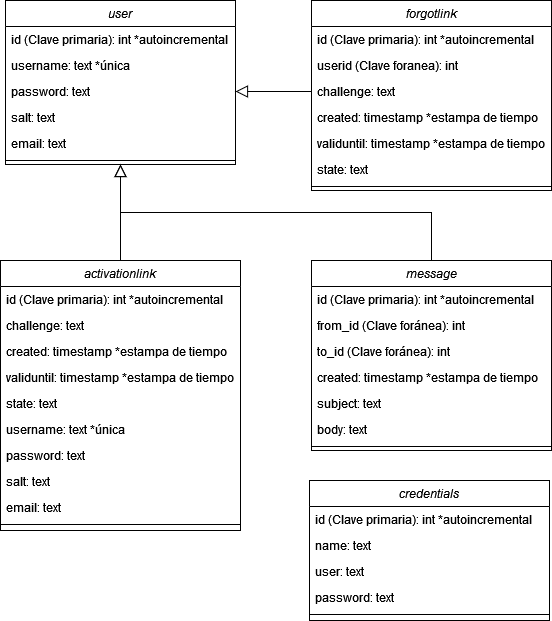
* Nombre método forgot ()
  + Ruta: / forgot
  + Métodos HTTP permitidos: ('POST')
  + Lógica algorítmica:

*El método forgot permite crear una entrada en la tabla forgotlink si el usuario invoca la función. El método registra el email del usuario y si es correcto selecciona el registro del usuario asociado a ese email en la tabla user. Luego genera un registro en la tabla forgotlink con el id del usuario, un número autogenerado hexadecimal y con estado de activo. Luego llama la función send\_email para enviar correo desde la cuenta principal de credentials al email del usuario con un link para cambiar la contraseña.*

* Nombre método login ()
  + Ruta: / login
  + Métodos HTTP permitidos: ('POST')
  + Lógica algorítmica:

*El método login permite acceder al inbox del usuario, por medio de la captura de los cambios de username y password el método verifica que ambos cambos estén llenados y compara contra la base de datos. Si es correcto almacena el id del usuario en la sesión y redirige al inbox del usuario. De lo contrario envia mesajes flash de error y redirige a login.html.*

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



Las tablas principales son *credentials*, *user y activationlink*, son aquellas que permiten establecer las funciones principales y conectar las otras tablas. Credentials maneja la cuenta base de almacenamiento que permite enviar correos de *autenticación* y *activación*. La tabla *user* es aquella que almacena todos los usuarios que se registren a la plataforma con un id, nombre usuario, contraseña, una semilla de encriptación y un email. La tabla *activationlink* permite almacenar un mensaje único a los y temporalmente los datos de usuarios que se registran por primera vez con un challenge y un salt. La tabla *forgotlink* permite almacenar un mesaje único cada vez que el usuario invoca la función de olvidar contraseña.

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

*El uso de método controlador establecido entre flask y python permite no mostrar el método de autenticación o manejo de información directamente en el código fuente de la página. Esto permite evitar tener brechas por posibles fallos en el código que puedan acceder terceros. La practica de implementación de métodos hash y semillas para encriptar contraseñas es mucho la forma mas avanzada para proteger el robo o hackeo de contraseñas. Las funciones hash son operaciones de encriptación irreversibles por lo que la base de datos nunca mostrará realmente los datos de contraseña. La aplicación de prepared statements para introducción de datos en las bases de datos SQL permite volver mas eficiente el código al enlazar las funciones de back-end y el cruce entre varias tablas de la base de datos.*