

Informe de avance de proyecto DreamsBank

Modulo Knowledge People

Grupo 4.2

11/05/2023

Equipo de trabajo:

Diana Lorena Balanta Solano

Carlos Javier Bolaños Riascos

Danna Alexandra Espinosa Arenas

George Michael Trujillo Riascos

Samuel Adrian Soto Ruiz

Carlos Manuel Villegas Ruiz

Universidad Icesi

Departamento TIC

Proyecto Integrador I

Introducción

En este informe del Sprint 2 del proyecto Knowledge People, se presentarán los avances y los resultados obtenidos en este periodo de trabajo. Como recordatorio, este proyecto tiene como objetivo principal crear un software que permita a los beneficiarios solicitar becas y a los donadores hacer donaciones de manera segura y efectiva.

En este Sprint, el equipo se ha enfocado en la implementación de los CRUDS de la aplicación. Para la realización de estas tareas se ha utilizado el framework Django y los lenguajes de programación Python, CSS y HTML.

Durante el Sprint, el equipo se enfrentó a algunos desafíos en cuanto a la comunicación y coordinación de tareas, lo que generó retrasos en la entrega de algunos elementos de la aplicación. Sin embargo, gracias a la colaboración y el compromiso de los miembros del equipo, se logró consolidar la comunicación y mejorar la planificación y distribución de tareas.

En este informe se presentarán las tareas realizadas durante el Sprint, los logros obtenidos y los desafíos enfrentados. También se detallarán los planes y objetivos para el próximo Sprint, con el fin de avanzar en el desarrollo del proyecto y lograr los objetivos establecidos.

1. Vista de entidades terminadas.

Durante el sprint anterior de nuestro proyecto Knowledge People, el equipo ha avanzado significativamente en la implementación de los crud del sistema. Se logró desarrollar y poner en marcha el crud de solicitud de becas así como el crud de usuario, entre otros, además se trabajó en terminar autenticación y se estableció trabajar con plantillas heredadas, las cuales también implementamos. Sin embargo, hubo algunos contratiempos y tareas pendientes que nos impidieron completar el desarrollo de todos los crud en el tiempo previsto.

Por esta razón, nos hemos propuesto como objetivo principal para el próximo sprint, finalizar los crud restantes del sistema. Esto incluye el crud de la gestión de becas por parte del rol administrador, así como el crud de gestión de donadores. Asimismo, continuaremos trabajando en las correcciones de errores y la mejora de la calidad de todo el sistema, asegurándonos de cumplir con los requisitos y expectativas de los usuarios.

Nuestro equipo está comprometido con el éxito de este proyecto y trabajará arduamente para completar estas tareas en el menor tiempo posible, sin

comprometer la calidad del software y manteniendo una comunicación efectiva en todo momento.

Las vistas de los crud's implementados se pueden visualizar en nuestro repositorio.

- [Repositorio GITHUB](#)

Adicionalmente, durante el último sprint del proyecto, hemos avanzado significativamente en el diseño de la interfaz de usuario del software Knowledge People. En particular, hemos creado prototipos de la interfaz para los roles de donante e institución. Estos prototipos se han creado después de un detallado análisis de las necesidades de cada uno de estos roles y de las funcionalidades que deben estar disponibles para ellos en la plataforma.

La creación de estos prototipos ha sido un paso importante en la implementación del software, ya que nos ha permitido visualizar la estructura y la organización de la información que se presentará en la interfaz de usuario. Además, estos prototipos han sido revisados y validados por el equipo y se han incorporado las sugerencias y comentarios para mejorar su usabilidad y eficiencia.

Para el próximo sprint, nuestro objetivo será continuar con el desarrollo de la interfaz de usuario para los diferentes roles.

- [Figma](#)

2. Pruebas unitarias e integración

En nuestro proyecto, hemos trabajado en la implementación de pruebas para garantizar la calidad del software. Para lograr esto, hemos definido los formatos de especificación de los escenarios y pruebas unitarias, así como el formato para las pruebas de integración.

Para los escenarios y pruebas unitarias, hemos definido un formato que incluye el nombre de la prueba, una descripción detallada del caso de prueba, los pasos a seguir para ejecutar la prueba, los resultados esperados y los resultados obtenidos. Este formato nos permite tener una especificación clara y detallada de cada prueba y nos ayuda a identificar y corregir errores en el código.

Para las pruebas de integración, hemos definido un formato que incluye la descripción de los componentes que se están integrando, los pasos para

ejecutar la prueba, los resultados esperados y los resultados obtenidos. Este formato nos permite asegurarnos de que todos los componentes se integren correctamente y que el software funcione de manera coherente en diferentes escenarios.

En resumen, la definición de estos formatos de especificación de pruebas nos ha permitido trabajar de manera más eficiente y efectiva en la implementación de pruebas en nuestro proyecto. Esto ha ayudado a mejorar la calidad del software y a garantizar que el software cumpla con los requisitos y expectativas del usuario final.

A continuación los links de acceso a nuestros formatos de pruebas:

- [Formato de escenarios y casos de prueba](#)
- [Formato de planteamiento de pruebas de integración](#)

¿Cómo hacer pruebas automáticas utilizando una base de datos alterna en Django?

En nuestro proyecto, hemos trabajado en la implementación de pruebas automáticas utilizando una base de datos alterna en Django. Para lograr esto, se definió una base de datos de prueba en el archivo de configuración de Django y se ejecutaron las pruebas utilizando esa base de datos.

Para crear la base de datos de prueba, Django automáticamente crea una base de datos separada para cada base de datos definida en la configuración de la aplicación. De esta manera, las pruebas se pueden ejecutar sin afectar la base de datos de producción.

Para definir la base de datos de prueba en Django, agregamos una entrada adicional en el diccionario DATABASES en el archivo settings.py. Esto nos permitió definir las credenciales de la base de datos de prueba y configurarla de manera adecuada. Luego, al ejecutar las pruebas, Django automáticamente utilizó la base de datos de prueba en lugar de la base de datos de producción.

Este enfoque ha demostrado ser muy efectivo para garantizar la calidad del software y para asegurarnos de que las pruebas no afecten la base de datos de producción. Además, nos ha permitido ahorrar tiempo y recursos al no tener que crear una base de datos de prueba separada manualmente. En general, estamos muy satisfechos con los resultados obtenidos y continuaremos utilizando este enfoque en nuestro proyecto.

```
# Database
# https://docs.djangoproject.com/en/4.2/ref/settings/
```

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    },
    'test': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
```

Definición de una base de datos alterna para hacer test.

Complicaciones:

Durante el desarrollo del proyecto, se presentaron diversas complicaciones que afectaron el avance del mismo. En primer lugar, el tiempo se convirtió en un factor crítico debido a la cantidad de tareas y objetivos a cumplir. Además, la falta de comunicación y coordinación en el grupo generó retrasos en la entrega de algunos elementos de la aplicación.

Por otro lado, también se presentaron complicaciones con el orden del flujo del proyecto, lo que afectó la planificación y distribución de tareas. A pesar de estos desafíos, el equipo trabajó en conjunto para superarlos y lograr avanzar en el proyecto de manera efectiva.

Objetivos para el próximo sprint:

Para el próximo sprint, nuestro equipo tiene varios objetivos para continuar mejorando nuestro proyecto de software Knowledge People. En primer lugar, haremos pruebas de usabilidad para asegurarnos de que la interfaz de usuario sea fácil de usar y atractiva para nuestros usuarios. También realizaremos simulaciones

de reportes de pagos para garantizar que el sistema maneje correctamente todas las transacciones de donaciones.

Además, nos enfocaremos en corregir los bugs que quedaron pendientes en el sprint anterior, así como en realizar pruebas de stress para asegurarnos de que el software sea confiable y robusto. Por supuesto, también nos aseguraremos de mejorar la comunicación en el equipo para garantizar una colaboración efectiva y eficiente.

Por último, nos aseguraremos de completar todas las tareas pendientes del sprint anterior para mantenernos en el camino hacia nuestros objetivos a largo plazo. Y todo esto lo haremos en el menor tiempo posible, manteniendo siempre la calidad y la eficacia en mente.

Enlaces Adjuntos:

A manera de recordatorio, el desarrollo de este proyecto y sus evidencias se encuentran extensamente en los siguientes enlaces, en donde se podrá encontrar:

1. Repositorio GitHub.

Por motivos de gestión y conflictos en el repositorio anterior, hemos decidido crear un nuevo repositorio y empezar de nuevo

[Repositorio nuevo de GitHub](#)

Donde se manejó el siguiente estándar de commits, commits tipo snake.
(estándar convencional de commits):

- *Tipo de commit:*
 - feat: La nueva característica que se agrega a una aplicación en particular
 - fix: Un parche para un error
 - style: Características o actualizaciones relacionadas con estilos

refactor: Refactorizar una sección específica de la base de código

test: Todo lo relacionado con pruebas

docs: Todo lo relacionado con documentación

chore: Mantenimiento de código regular.

- El título separado del cuerpo del mensaje por una línea en blanco
- Quitar signos de puntuación innecesarios
- No terminar el título con un punto
- Usar mayúscula al inicio del título
- Usar el modo imperativo en el título (*verbo en tiempo presente*).

Ejemplo de estándar de commit:

docs_Add_the_README.md file

2. Formato de escenarios y casos de prueba

[Formato de escenarios y casos de prueba](#)

3. Escenarios y casos de prueba de integración:

[Formato de planteamiento de pruebas de integración](#)

4. Informe de dailys:

[Informe dailys sprint 2](#)

5. Figma del proyecto:

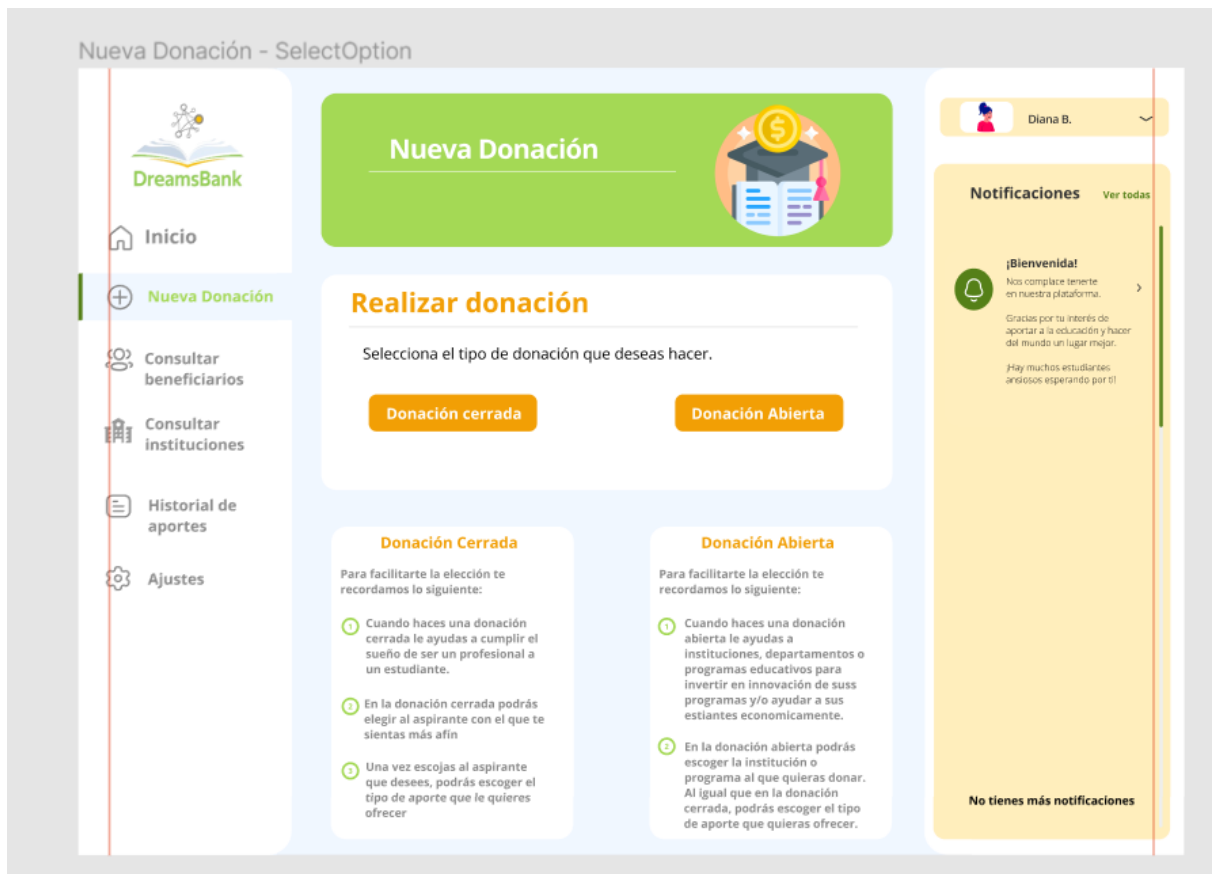
[Figma](#)

6. Tablero de Jira

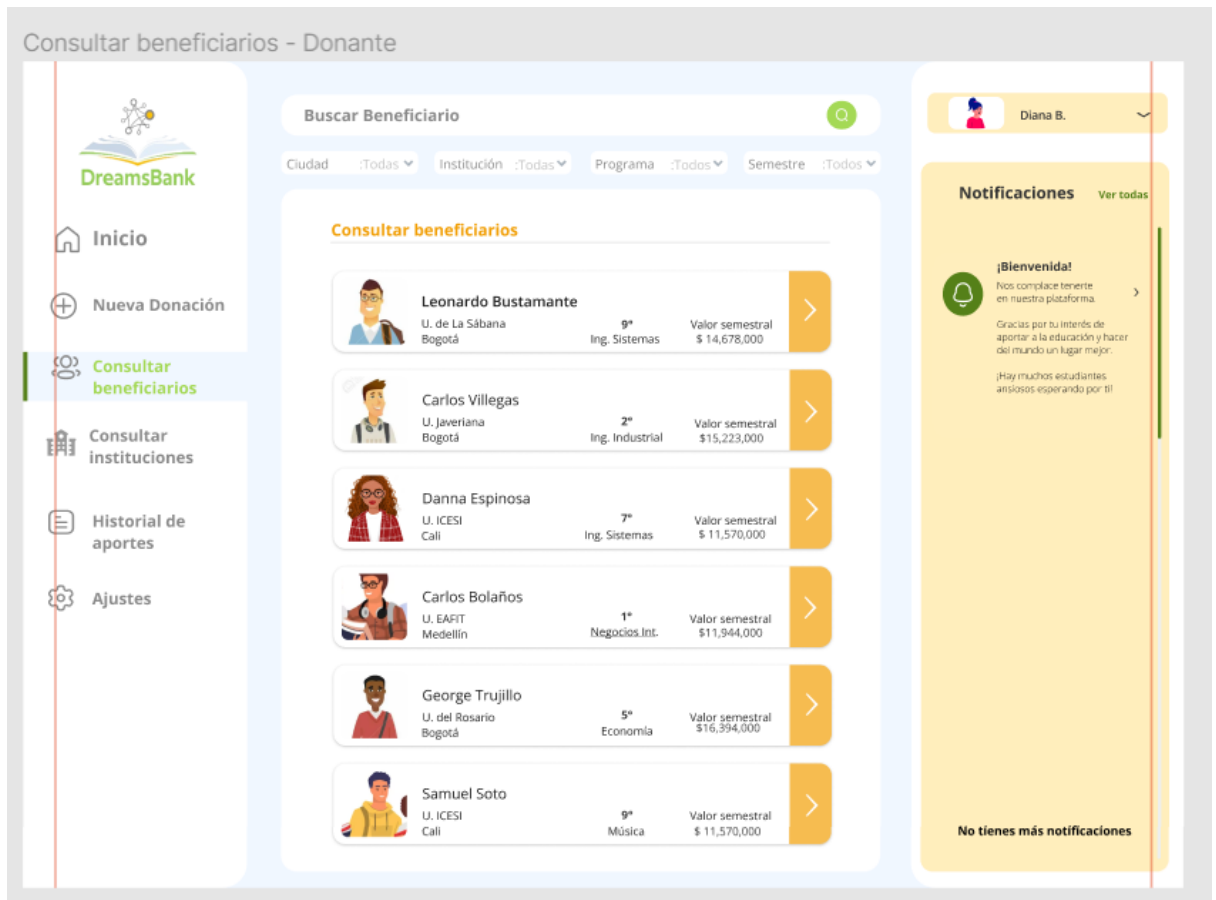
[Tablero](#)

7. Pantallazos, vistas y avances.

Figma:



Menú principal de los usuario con rol Donante.



Posibles beneficiarios

Autenticación:



127.0.0.1:8000/users/signin/

Repositorios LaTeX algebra - @Q... Estandar commits Jira-PI Tips markdown LATEX _ Importante Database Systems... Shazam: identifica...

DreamsBank Nosotros Aliados Contacto Registrarse Iniciar Sesión

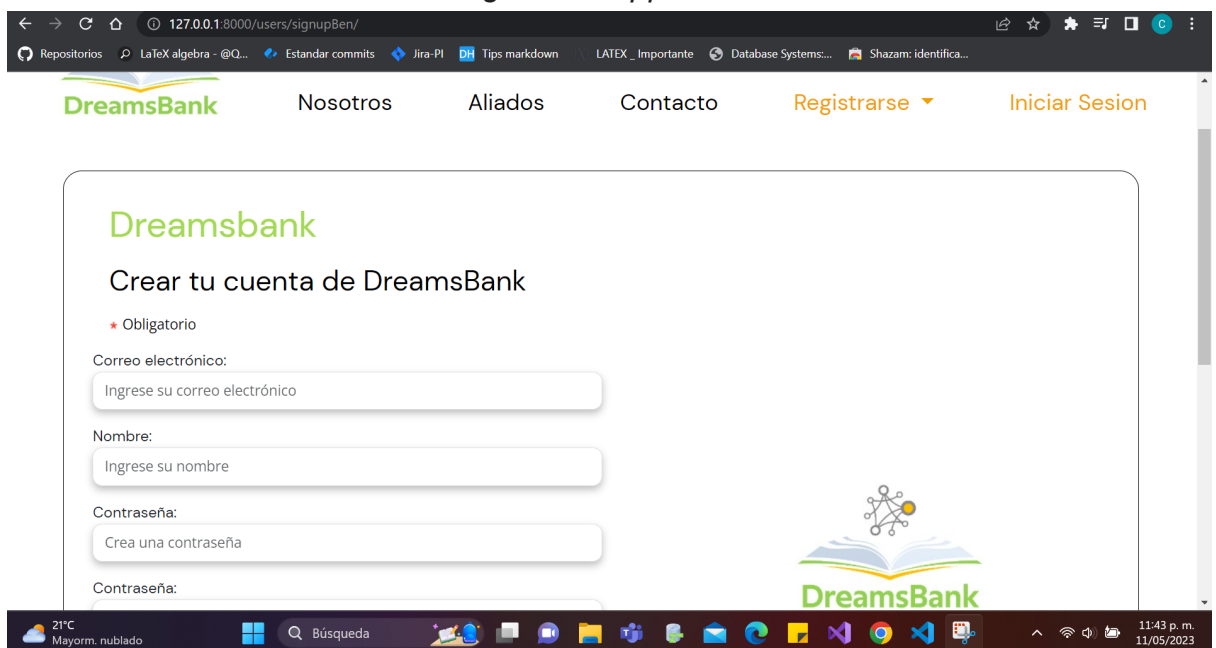
Inicio de Sesión

Correo electrónico

Contraseña

Login

Login de la app



127.0.0.1:8000/users/signupBen/

Repositorios LaTeX algebra - @Q... Estandar commits Jira-PI Tips markdown LATEX _ Importante Database Systems... Shazam: identifica...

DreamsBank Nosotros Aliados Contacto Registrarse Iniciar Sesión

Dreamsbank

Crear tu cuenta de DreamsBank

* Obligatorio

Correo electrónico:

Ingrese su correo electrónico

Nombre:

Ingrese su nombre

Contraseña:

Crea una contraseña

Contraseña:

DreamsBank

21°C Mayorm. nublado Búsqueda 11:43 p. m. 11/05/2023

Formulario de registro

Formulario de solicitud:

The screenshot shows a web browser at the URL 127.0.0.1:8000/scholarships/newapplication/. The page has a green header with the title "Formulario de solicitud" and a logo of a graduation cap. On the left is a sidebar with the DreamsBank logo and navigation links: Inicio, Nueva solicitud de beca, Historial de donantes, Historial de solicitudes, and Solicitud activa. The main content area is titled "Complete sus datos" and contains several input fields: Nombres, Apellidos, Tipo de documento (a dropdown menu), Número de documento, Institución (a dropdown menu), Programa, Valor del periodo, and Periodo actual. On the right, there is a user profile for "noima Carlitos12" and a notification box with a welcome message: "¡Bienvenida! Nos complace tenerte en nuestra plataforma. Esperamos que tu ingreso sea beneficioso para tí. Siempre estamos para ayudarte en tu formación académica."

Formulario solicitud

Pruebas unitarias:

The screenshot shows the Visual Studio Code editor with a file named tests.py open. The file contains a Python class UserTest that inherits from unittest.TestCase. It has a setUp method that creates a User object and returns it. There are two test methods: test_scenario_1, which calls setUp and prints an error message if it fails, and test_scenario_2, which sets specific attributes for a User object (email, password, name, idType, numID, role) and then asserts that the email is 'john@example.com'. The Explorer sidebar on the left shows a project structure with files like migrations, templates, __init__.py, admin.py, apps.py, forms.py, managers.py, models.py, tests.py, urls.py, views.py, and a venv directory. The status bar at the bottom indicates the file is at line 40, column 1, using UTF-8 encoding with CRLF line endings, and is a Python 3.11.2 file in a virtual environment.

```
class UserTest(TestCase):
    def setUp(self):
        user = User()
        return user

    def test_scenario_1(self):
        try:
            user = self.setUp()
        except:
            print("User object didn't create")

    def test_scenario_2(self):
        user = self.setUp()
        user.email = "john@example.com"
        user.password = "johndoe1"
        user.name = "John Doe"
        user.idType = 1
        user.numID = "112508374"
        user.role = 1

        self.assertEqual(user.email, "john@example.com")
```