

Universidad de San Carlos de Guatemala  
Facultad de Ingeniería  
Estructura de Datos  
Sección B  
Ing. Carlos Alonzo  
Aux. Marco Galindo



## PROYECTO #1

### Renta de Activos

#### Objetivos

- Entender y lograr implementar estructuras tipo árbol.
- Afinar los conocimientos de la matriz dispersa.
- Implementar servicios web.
- Implementar API's.
- Crear aplicaciones android.

#### Descripción

Se le solicita a los estudiantes realizar un sistema de renta de activos entre empresas. Dentro del sistema, se manejarán rentas, usuarios, un historial de transacciones y reportes. Todo el sistema deberá de poder ser controlado mediante una aplicación android.

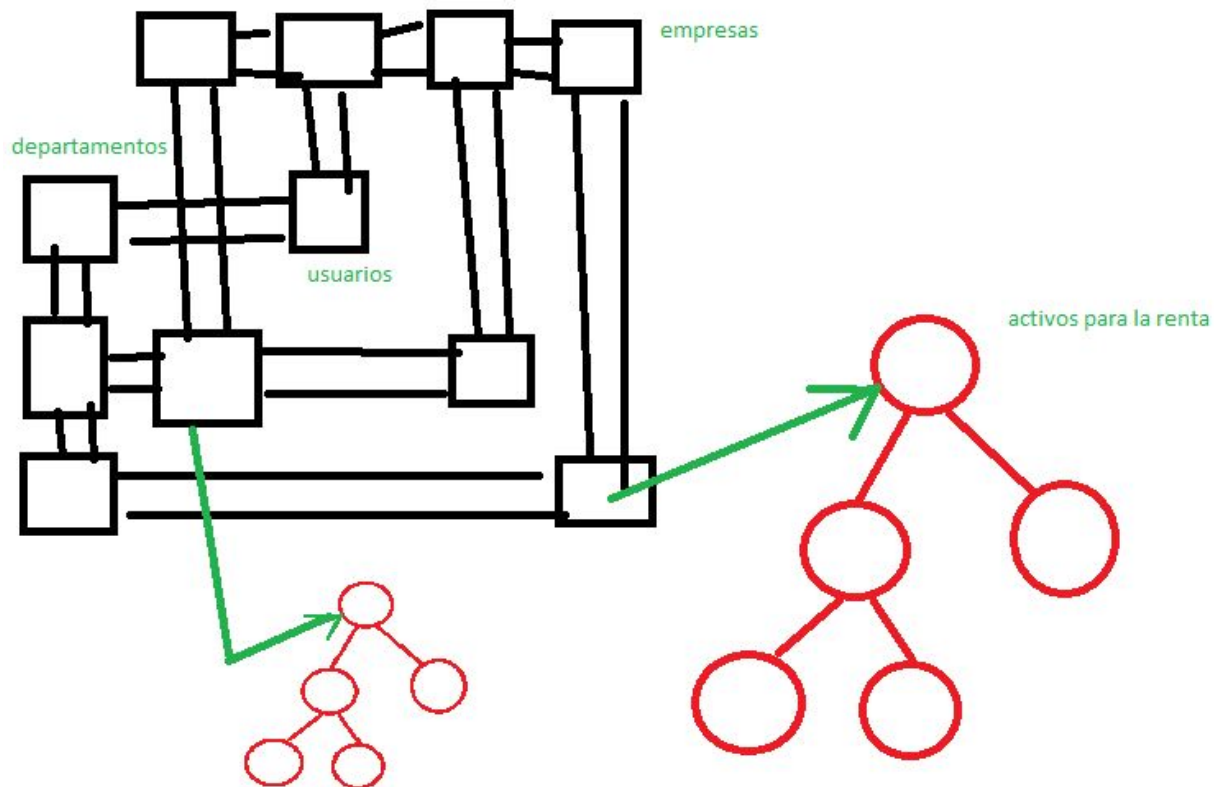
#### Implementación

Se deberá de implementar un servicio web codificado en **Python** utilizando el framework **Flask** para la parte que maneje usuarios y catálogos de activos disponibles para la renta. Los encabezados horizontales de la matriz dispersa serán nombres de distintas empresas, los encabezados verticales serán los diferentes departamentos que pueden existir dentro de una empresa (ventas, mercadeo, etc...) y el nodo de la matriz contendrá nombres de personas que serán los encargados dentro de la organización para realizar la renta de los activos. Cada nodo de la matriz dispersa deberá de contener al usuario, con su nombre de usuario (para realizar su login), contraseña y nombre completo, así como un árbol binario de búsqueda equilibrado (AVL) donde se almacenarán los activos que el usuario ponga a la renta. Los activos van a tener un nombre, una descripción y un ID **único para cada activo** que será un código alfanumérico de 15 caracteres, por ejemplo "tb86vw4rh194hpk".

NOTA: El nombre de usuario se podrá repetir siempre y cuando no se repita en un solo departamento y empresa.

Estructuras a crear en Python:

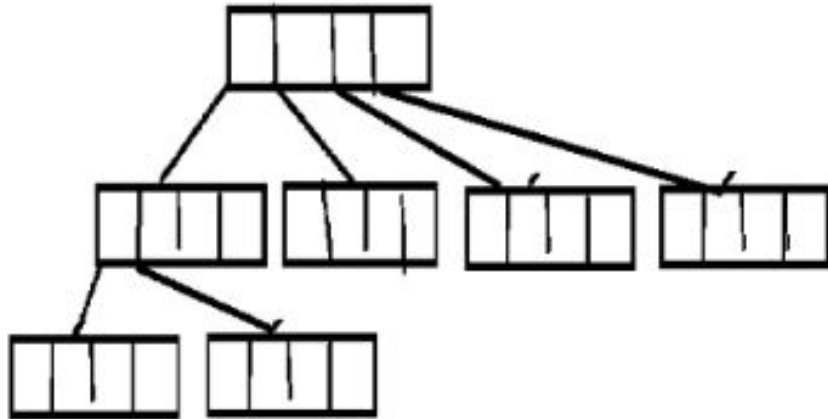
- Matriz dispersa
- AVL



Adicionalmente a esto, se deberá de crear una API en **asp.net** junto a **C#** que maneje la parte de los historiales de transacciones. Para esto se tendrá un árbol B **de orden 5** donde cada nodo contendrá toda la información de las transacciones que se han realizado en el sistema. Cada vez que se rente un activo, se deberá de crear una nueva transacción, esta tendrá la siguiente información:

- Un ID único para la transacción (un código alfanumérico de 15 caracteres)
- El ID del activo que se rentó
- El usuario que realizó la renta
- La empresa y departamento pertenecientes al usuario que realizó la renta
- La fecha en que se realizó la renta
- Por cuánto tiempo se rentó

El árbol B deberá estar ordenado mediante el ID único de cada transacción.



Estructuras a crear en C#:

- Arbol B

El sistema podrá ser controlado a través de una aplicación creada en **android** así como desde una interfaz gráfica web creada en **jsp**.

### Interfaz Web (JSP)

Se deberá crear una interfaz web con JSP y Java que administre la creación de usuarios y creación de catálogos de productos pertenecientes a una empresa bajo la responsabilidad del cliente que creó la oferta de renta. Los usuarios que se creen mediante esta aplicación deberán de ser guardados en la matriz dispersa de Python y los activos que esté agregue al sistema deben de guardarse en su AVL.

Debe de existir una opción de modificar la descripción de un activo que esté a la renta, así como la opción de eliminar un activo de su AVL.

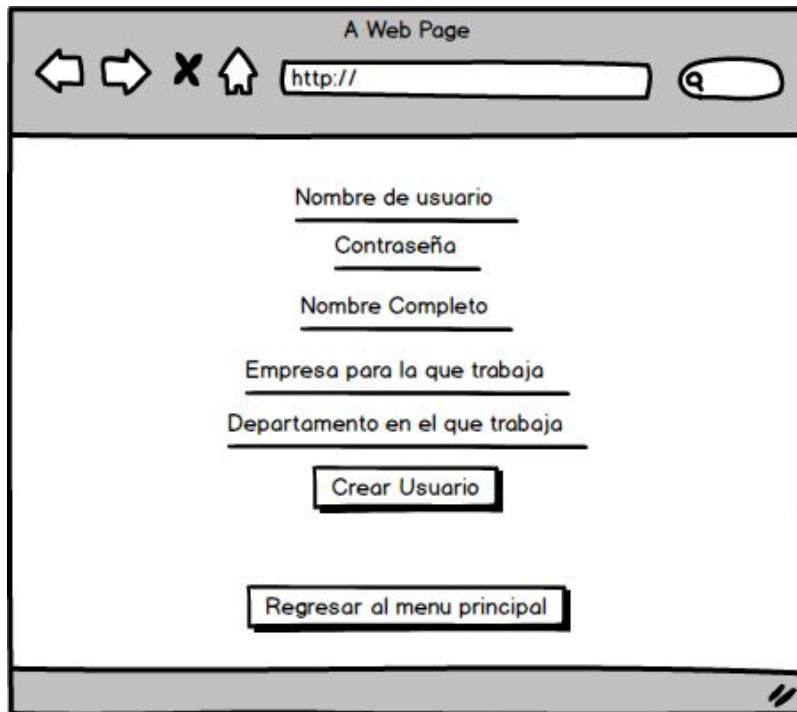
## Inicio de Sesión (JSP)

A hand-drawn diagram of a web browser window. The title bar at the top is labeled "A Web Page". Below the title bar is a navigation bar containing four icons: a left arrow, a right arrow, a close button (X), and a home button (house). To the right of these icons is a text input field containing "http://" and a search button (magnifying glass). The main content area of the browser contains a login form with the following elements:

- Usuario
- Contraseña
- Empresa
- Departamento
- 
- 

The bottom of the browser window has a gray status bar with a small icon in the bottom right corner.

## Creación de usuarios (JSP)



A Web Page

Navigation icons: back, forward, stop, home, search.

Address bar: http://

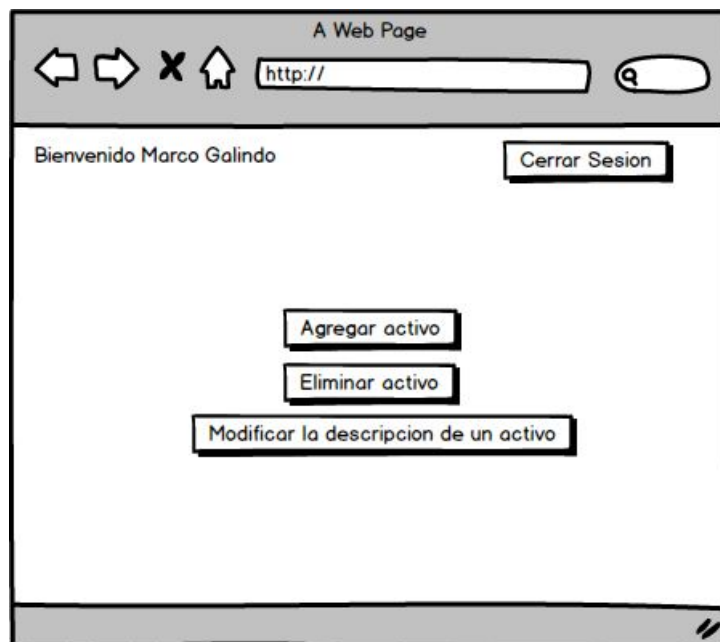
Form fields:

- Nombre de usuario
- Contraseña
- Nombre Completo
- Empresa para la que trabaja
- Departamento en el que trabaja

Buttons:

- Crear Usuario
- Regresar al menu principal

## Menú principal de un usuario ya ingresado



A Web Page

Navigation icons: back, forward, stop, home, search.

Address bar: http://

Header:

Bienvenido Marco Galindo

Buttons:

- Cerrar Sesión
- Agregar activo
- Eliminar activo
- Modificar la descripción de un activo

## Creación de activos (JSP)

Al ingresar la opción "Agregar activo" el ID del nuevo activo se crea automática y aleatoriamente.

The screenshot shows a web browser window with the title "A Web Page". The address bar contains "http://". The page content includes a welcome message "Bienvenido Marco Galindo" and a button labeled "Regresar al menu". Below this is a form with a label "Nombre" above a text input field. Underneath the text field is a large text area labeled "Descripcion". At the bottom of the form is a button labeled "Agregar activo".

## Eliminación de activos (JSP)

Los campos de nombre y descripción se tienen que llenar automáticamente con la información correspondiente al ID seleccionado del ComboBox. El comboBox debe tener todos los IDs existentes a cargo del usuario que inició sesión.

A hand-drawn mockup of a web browser window titled "A Web Page". The browser's address bar shows "http://". The page content includes a welcome message "Bienvenido Marco Galindo" and a "Regresar al menu" button. Below this is a section titled "ID del Producto" containing a dropdown menu with the selected value "3tvesv56d1gg86r". Under the dropdown is a label "Nombre" above a large text input field labeled "Descripcion". At the bottom of this section is a button labeled "Eliminar activo".

## Modificación de activos (JSP)

Los campos de nombre y descripción se tienen que llenar automáticamente con la información correspondiente al ID seleccionado del ComboBox. El campo de descripción se puede modificar y el cuando se escoja la opción “modificar la descripción”, la nueva descripción será la que esté en el área de texto en ese momento. El comboBox debe tener todos los IDs existentes a cargo del usuario que inició sesión.

A Web Page

http://

Bienvenido Marco Galindo

Regresar al menu

ID del Producto

3tvesv56d1gg86r

Nombre

Descripcion

Modificar la descripcion

**NOTA:** La interfaz web debe de ser responsiva en cualquier dispositivo, para esto se debe de implementar la herramienta **Bootstrap**.



## Aplicación Android y Renta de Activos

Se deberá de crear una app de Android que maneje la parte de la renta de los activos. Se debe crear un login de usuarios, y luego de estar en la sesión de cierto usuario, se deberá de escoger en un catálogo de activos, el activo que se desea alquilar y el tiempo de alquiler. Un activo debe ser eliminado del catálogo una vez se realiza una renta sobre el.

También deberá de existir un módulo donde se registre la devolución del activo. Esto regresará el activo al catálogo de activos disponibles.

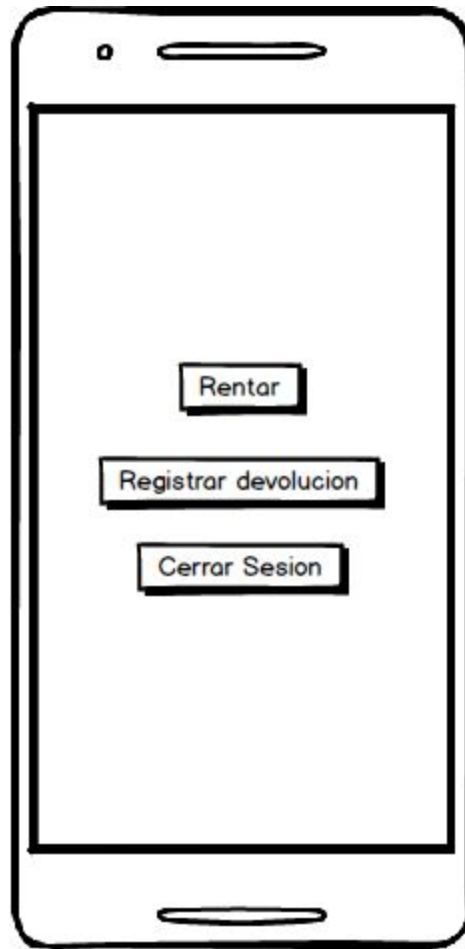
**NOTA:** Si un activo es eliminado mediante la aplicación web en JSP, todas las transacciones en el árbol B que incluyan a este activo deberán de ser eliminadas también.

### Inicio de Sesión (Android)

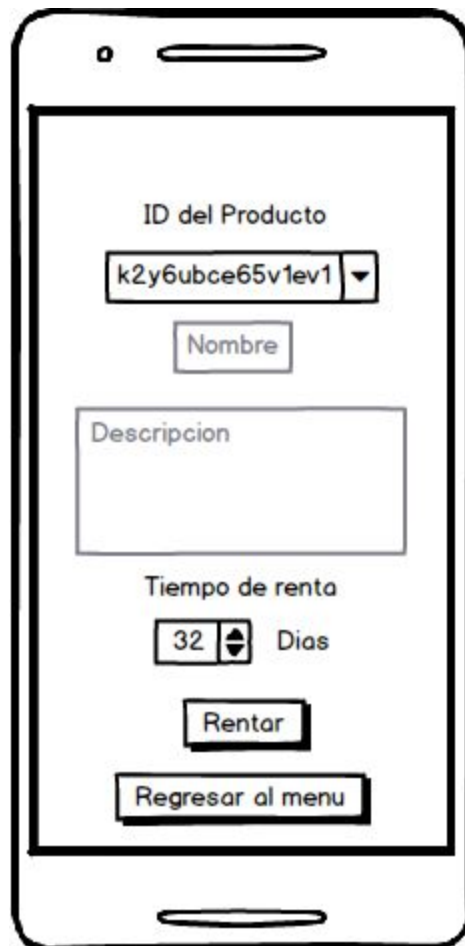
Diagrama de la pantalla de inicio de sesión de una aplicación Android. La pantalla muestra los siguientes campos de entrada y botones:

- Usuario:
- Contraseña:
- Empresa:
- Departamento:
- Botón:

## Menu principal (Android)



## Renta de activos (Android)



A mobile application interface for asset rental, displayed on a smartphone screen. The interface includes a header bar with a back arrow and a title. The main content area contains a form with the following elements: a label 'ID del Producto' above a dropdown menu showing 'k2y6ubce65v1ev1'; a label 'Nombre' above a text input field; a label 'Descripcion' above a larger text input field; a label 'Tiempo de renta' above a numeric input field showing '32' and a unit selector 'Dias'; a 'Rentar' button; and a 'Regresar al menu' button at the bottom.

ID del Producto

k2y6ubce65v1ev1 ▼

Nombre

Descripcion

Tiempo de renta

32 ▴ ▾ Dias

Rentar

Regresar al menu

## Registro de devoluciones (Android)

El comboBox deberá de mostrar todos los IDs de los artículos que el usuario rento, es decir, son artículos temporalmente en la posesión del usuario que inicio sesión. Ningún dato podrá ser modificado en esta actividad de la aplicación.

The image shows a mockup of an Android application screen. At the top, there is a header bar with a small circular icon on the left and a horizontal line in the center. Below the header, the screen displays the following elements: a label 'ID del Producto' above a dropdown menu showing 'k2y6ubce65v1ev1' with a downward arrow; a label 'Nombre' above a text input field; a label 'Descripcion' above a larger text input field; a label 'Tiempo de renta' above the text '32 Dias'; a button labeled 'Regresar'; and a button labeled 'Regresar al menu' at the bottom. The entire screen is enclosed in a rounded rectangle representing a smartphone.

## Flujo de la aplicación

Básicamente el sistema tendrá el siguiente flujo:

- Se crearán usuarios dentro del sistema con contraseña para poder realizar su login.
- Este usuario va a pertenecer a una empresa y un departamento de la empresa.
- El usuario podrá poner activos de la empresa para la renta.
- Otros usuarios podrán rentar esos activos.

## Reportes

Se deberá crear una aplicación sencilla de escritorio de Java donde se van a mostrar los siguientes reportes:

- El estado actual de la matriz dispersa
- Todos los activos pertenecientes a una empresa que esten disponibles para rentarse.
- Todos los activos pertenecientes a un departamento en específico que estén disponibles para rentarse.
- El árbol B
- El AVL de cierto usuario
- Todos los activos que un usuario tenga en su posesión actualmente (activos que el haya rentado)

### Aplicación de Reportes (Java)

Reportes

Matriz Dispersa

Nombre de una empresa

Activos de una empresa

Nombre de un departamento

Activos de un departamento

Arbol B

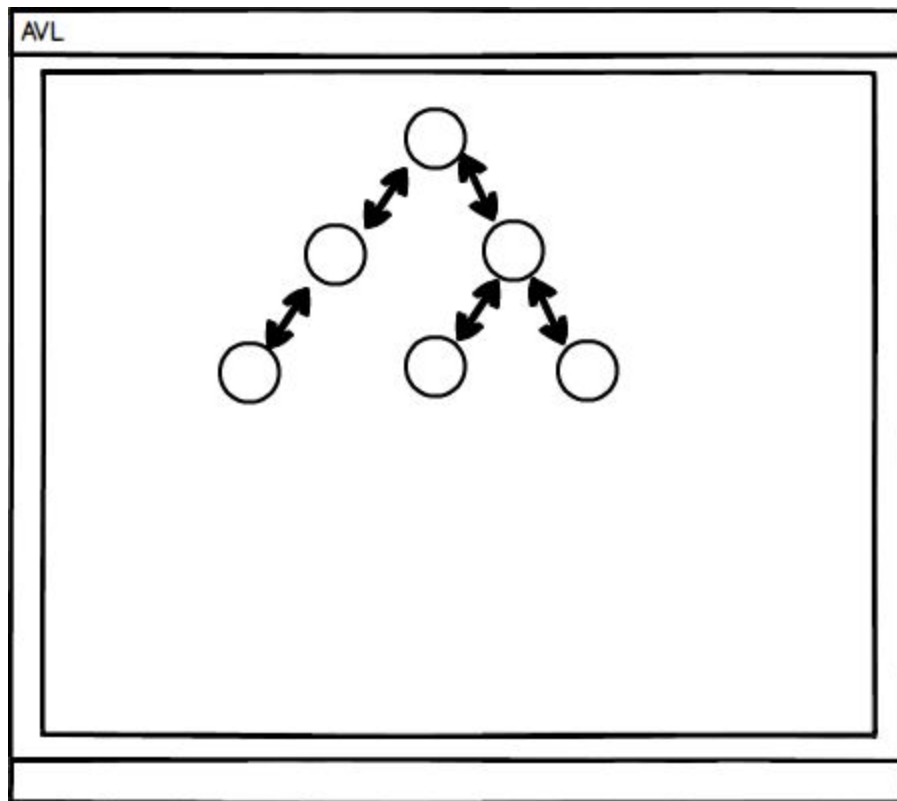
Usuario3 ▼ Empresa1 ▼ Departamento1 ▼

AVL del Usuario seleccionado

Usuario3 ▼ Empresa1 ▼ Departamento1 ▼

Activos rentados por el usuario seleccionado

Cuando se seleccione un reporte este debe ser mostrado en otra ventana. Por ejemplo:



## Restricciones

Los reportes son esenciales para verificar si se trabajó correctamente la estructura, así que **sin reportes, no se calificará el proyecto.**

## Penalizaciones

Si durante la calificación del proyecto, el programa arroja un **NullPointerException**, se restará el 25% de la nota obtenida por cada vez que salga el error.

## Observaciones

- La calificación se realizará utilizando varias de las computadoras del grupo, por ejemplo, se podrá correr un servidor en la computadora1, otro servidor en la computadora2, la aplicación JSP en la computadora3 y se utilizara algún dispositivo android para la aplicación.

- Las estructuras deben de ser realizadas por el estudiante. No está permitido el uso de ArrayLists, librerías de colas, librerías de matrices ni de arreglos estáticos ni de árboles.
- Durante la calificación, se harán preguntas a estudiantes al azar.
- Lenguaje a utilizar: Java, Python, asp.net, JSP, los previos lenguajes con cualquier IDE y Android utilizando Android Studio.
- El proyecto será realizado en grupos de 4 integrantes.
- Una persona del grupo debe de crear un repositorio en GitHub con el nombre de Proyecto1s12017\_#carnet1\_#carnet2\_#carnet3\_#carnet4.
- Los documentos deben estar sincronizados con GitHub así mismo teniendo un historial de que trabajo en dicha plataforma
- Forma de entrega: vía GitHub
- Fecha de Entrega: 28 de Marzo del 2017 a las 11:59 P.M. (Última sincronización de GitHub).
- **COPIAS SERÁN PENALIZADAS con una nota de 0.**
- **COPIAS SERÁN PENALIZADAS conforme al reglamento.**