Informe sobre el libro

las riquezas de las naciones adam smith

cap 8

jhoan bernal

13 de julio de 2025

Introducción

El Capítulo 8 de "La Riqueza de las Naciones" de Adam Smith aborda los fundamentos de los salarios del trabajo, diferenciando entre el salario natural y el salario de mercado, y analizando cómo factores como la oferta, la demanda, la producción y las condiciones económicas influyen en la remuneración de los trabajadores. Inspirados en estos principios, este informe documenta el desarrollo de una simulación interactiva que permite experimentar, con datos reales y visualizaciones modernas, los mecanismos económicos descritos por Smith.A lo largo del informe se detallan los siguientes puntos clave:

- Obtención y procesamiento de datos reales:

Integración con la API de USDA para obtener información actualizada sobre la producción agrícola en distintos estados y años.

- Simulación del salario ajustado:

Cálculo dinámico del salario de los trabajadores agrícolas en función del valor de la producción, emulando el concepto de salario de mercado y su dependencia de la riqueza generada.

- Modelado de la oferta laboral:

Relación directa entre el volumen de producción y la cantidad de ofertas laborales, reflejando cómo la actividad económica determina la demanda de trabajo.

- Análisis de la demanda laboral a lo largo del tiempo:

Visualización de la evolución de la demanda de empleo agrícola en diferentes años, permitiendo identificar tendencias y el impacto de factores externos.

- Visualización y experiencia de usuario:

Presentación de los resultados mediante tablas, gráficos interactivos y acceso transparente a los datos crudos, facilitando la comprensión y el análisis.

Esta simulación no solo ilustra los conceptos teóricos del Capítulo 8, sino que los lleva a la práctica, permitiendo al usuario explorar y entender cómo los salarios y el mercado laboral responden a los cambios en la producción y la economía real.

1. Objetivo

Integrar la API pública de USDA (Departamento de Agricultura de EE.UU.) en el proyecto para obtener datos agrícolas reales y utilizarlos en simulaciones y visualizaciones relacionadas con el Capítulo 8 del libro "La Riqueza de las Naciones".

2. Pasos Realizados

A. Creación de un Paquete Reutilizable para la API de USDA

Archivo: internal/usdaapi/usdaapi.go

- Se creó un paquete llamado `usdaapi` para encapsular la lógica de consulta a la API de USDA.

- Se implementó la función `GetUSDAData`, que recibe parámetros como commodity, año, estado y categoría estadística, y retorna los datos obtenidos de la API como una lista de mapas.

- El código incluye comentarios explicativos en cada paso clave: construcción de la URL, petición HTTP, lectura y parseo del JSON, y validación de la respuesta.

```
package usdaapi


import (
    "encoding/json"
    "fmt"
    "io"
    "net/http"
)


// USDAResponse representa la estructura de la respuesta de la
API de USDA
```

```
// El campo Data contiene los registros devueltos por la consulta
// Cada registro es un mapa de clave-valor
type USDAResponse struct {
    Data []map[string]interface{} `json:"data"`
}

// GetUSDAData consulta la API de USDA con los parámetros dados y retorna los datos
func GetUSDAData(commodity, year, state, statisticcat string) ([]map[string]interface{}, error) {
    url := fmt.Sprintf(

"https://quickstats.nass.usda.gov/api/api_GET/?key=1F325726-42E7-3E08-8E7F-C7ED7047890A&commodity_desc=%s&year=%s&state_alpha=%s&statisticcat_desc=%s&format=JSON",
        commodity, year, state, statisticcat,
    )

    // Hacer la petición HTTP GET
    resp, err := http.Get(url)
    if err != nil {
```

```
        return nil, fmt.Errorf("error al hacer la petición: %w",
err)
    }
    defer resp.Body.Close()

    // Leer el cuerpo de la respuesta
    body, err := io.ReadAll(resp.Body)
    if err != nil {
        return nil, fmt.Errorf("error leyendo el cuerpo de la
respuesta: %w", err)
    }

    // Parsear el JSON
    var parsed USDAResponse
    if err := json.Unmarshal(body, &parsed); err != nil {
        return nil, fmt.Errorf("error parseando JSON: %w", err)
    }

    // Validar que haya datos
    if len(parsed.Data) == 0 {
        return nil, fmt.Errorf("la respuesta no contiene datos")
```

```
    }


    return parsed.Data, nil
}
```

3. Creación de un Comando de Ejemplo para Visualizar los Datos

Archivo: cmd/usda_example/main.go

- Se creó un ejecutable de ejemplo que utiliza el paquete `usdaapi` para consultar la API de USDA.

- El comando define parámetros de ejemplo (maíz, año 2023, Iowa, producción) y llama a la función de consulta.

- Muestra en consola la cantidad de registros obtenidos y un ejemplo de registro, imprimiendo cada campo y su valor.

- El código está documentado con comentarios para facilitar su comprensión y reutilización.

Fragmento relevante:

```
package main


import (
    "fmt"
    "log"
```

```
"github.com/jhoan28310576/cap7-8-9_las_riquesas_de_las_naciones/internal/usdaapi"
)

func main() {
    // Parámetros de ejemplo: maíz, año 2023, Iowa, producción
    commodity := "CORN"
    year := "2023"
    state := "IA"
    statisticcat := "PRODUCTION"

    // Llamar a la función que consulta la API de USDA
    data, err := usdaapi.GetUSDAData(commodity, year, state, statisticcat)
    if err != nil {
        log.Fatalf("Error obteniendo datos de USDA: %v", err)
    }

    // Mostrar la cantidad de registros y un ejemplo
    fmt.Printf("Se obtuvieron %d registros de la API de USDA.\n", len(data))
```

```
    if len(data) > 0 {
        fmt.Printf("Ejemplo de registro:\n")
        for k, v := range data[0] {
            fmt.Printf("  %s: %v\n", k, v)
        }
    }
}
```

// Llamar a la función que consulta la API de USDA

```
data, err := usdaapi.GetUSDAData(commodity, year, state,
statisticcat)
```

// ... manejo de errores ...

```
    // Mostrar la cantidad de registros y un ejemplo
    fmt.Printf("Se obtuvieron %d registros de la API de
USDA.\n", len(data))
    if len(data) > 0 {
        fmt.Printf("Ejemplo de registro:\n")
        for k, v := range data[0] {
            fmt.Printf("  %s: %v\n", k, v)
```

```
        }

    }

}
```

C. Ejecución y Resultado

```
PS C:\Users\jhoan\Desktop\cap7,8,9_las riquesas de las naciones>
go run cmd/usda_example/main.go

Se obtuvieron 115 registros de la API de USDA.

Ejemplo de registro:

  short_desc: CORN, GRAIN - PRODUCTION, MEASURED IN $

  watershed_code: 00000000

  county_ansi:

  domain_desc: TOTAL

  asd_code:

  congr_district_code:

  state_ansi: 19

  reference_period_desc: YEAR

  source_desc: SURVEY

  year: 2023

  Value: 11,628,956,000
```

```
end_code: 00

group_desc: FIELD CROPS

statisticcat_desc: PRODUCTION

load_time: 2025-02-26 15:00:00.000

sector_desc: CROPS

location_desc: IOWA

asd_desc:

zip_5:

CV (%):

week_ending:

domaincat_desc: NOT SPECIFIED

agg_level_desc: STATE

prodn_practice_desc: ALL PRODUCTION PRACTICES

region_desc:

freq_desc: ANNUAL

class_desc: ALL CLASSES

util_practice_desc: GRAIN

begin_code: 00

country_code: 9000

watershed_desc:

county_name:
```

```
  state_alpha: IA

  commodity_desc: CORN

  state_fips_code: 19

  state_name: IOWA

  country_name: UNITED STATES

  county_code:

  unit_desc: $
```

Se ejecutó el comando con:

terminal:

go run cmd/usda_example/main.go

```

**Resultado en consola:**

```

Se obtuvieron 115 registros de la API de USDA.

Ejemplo de registro:

```
sector_desc: CROPS

short_desc: CORN, GRAIN - PRODUCTION, MEASURED IN $

util_practice_desc: GRAIN

domaincat_desc: NOT SPECIFIED
```

```
...
state_name: IOWA
unit_desc: $
year: 2023
```

Esto confirma que la integración es exitosa y que los datos reales de la API de USDA están disponibles para ser usados en simulaciones, visualizaciones o análisis.

- Crear un endpoint en Go para exponer estos datos a una página web.

4.4.1 Implementación del Endpoint

Se creó un endpoint RESTful en el backend usando Gin para exponer la simulación del mercado laboral basada en datos reales de la USDA Quick Stats API:

- Ruta: /api/cap8/simulacion
- Método: GET
- Parámetros de consulta:
- anio (opcional, por defecto 2023)
- estado (opcional, por defecto IA)
- Respuesta: JSON con los siguientes campos:
- salarioAjustado: Salario simulado ajustado por producción real
- valorProduccion: Valor de producción agrícola real
- numOfertas: Número de ofertas laborales simuladas
- demandaAnual: Array con demanda laboral por año
- apiRaw1, apiRaw2, apiRawDemanda: Datos crudos de la API para transparencia y depuración

fragmento de codigo mas relevantes:

```
// 3. Demanda laboral ajustada por año

anio := c.Query("anio")

if anio == "" { anio = "2023" }

estado := c.Query("estado")

if estado == "" { estado = "IA" }

data, err := usdaapi.GetUSDAData("CORN", anio, estado,
"PRODUCTION")

// ... procesamiento ...

c.JSON(http.StatusOK, gin.H{

    "salarioAjustado": salarioAjustado,

    "valorProduccion": valorProduccion,

    "numOfertas": numOfertas,

    "demandaAnual": demandaAnual,

    "apiRaw1": data,

    "apiRaw2": data,

    "apiRawDemanda": apiRawDemanda,

})
```

4.5 Procesamiento y Lógica de Simulación

4.5.1 Obtención y Procesamiento de Datos

- Se consulta la API de USDA para obtener el valor de producción de maíz según el estado y año seleccionados.
- Se calcula el salario ajustado y el número de ofertas laborales en función de la producción real.
- Se simula la demanda laboral para varios años, permitiendo observar tendencias históricas.

Algoritmo de simulación:

```
// Salario ajustado
salarioAjustado := salarioBase
if valorProduccion > 0 {
    salarioAjustado += valorProduccion / 1e10
}
// Ofertas laborales
numOfertas := 2
if valorProduccion > 0 {
    numOfertas += int(valorProduccion / 1e10)
}
// Demanda anual
for _, anioConsulta := range anios {
    // ... obtener valorProdAnio ...
    numDemandas := 2
```

```
    if valorProdAnio > 1e10 { numDemandas += 1 }

    demandaAnual = append(demandaAnual, ...)

}
```

Ejecución y test usda_handlers.go de respuesta de la api USDA [USDA - National Agricultural Statistics Service - Quick Stats]

resultado api test terminal raiz proyecto :

```
PS C:\ go run cmd/usda_example/main.go riquesas de las naciones>

Se obtuvieron 115 registros de la API de USDA.

Ejemplo de registro:

  asd_code:

  county_code:

  source_desc: SURVEY

  freq_desc: ANNUAL

  short_desc: CORN, GRAIN - PRODUCTION, MEASURED IN $

  region_desc:

  domain_desc: TOTAL

  zip_5:

  reference_period_desc: YEAR

  end_code: 00

  county_name:

  statisticcat_desc: PRODUCTION
```

```
begin_code: 00

Value: 11,628,956,000

unit_desc: $

group_desc: FIELD CROPS

watershed_desc:

county_ansi:

state_fips_code: 19

asd_desc:

year: 2023

prodn_practice_desc: ALL PRODUCTION PRACTICES

load_time: 2025-02-26 15:00:00.000

week_ending:

sector_desc: CROPS

state_alpha: IA

country_code: 9000

util_practice_desc: GRAIN

CV (%):

agg_level_desc: STATE

state_ansi: 19

congr_district_code:

country_name: UNITED STATES
```

```
domaincat_desc: NOT SPECIFIED

watershed_code: 00000000

class_desc: ALL CLASSES

state_name: IOWA

location_desc: IOWA

commodity_desc: CORN
```

4.5.2 Manejo de Parámetros y Validaciones

- El endpoint valida la presencia de los parámetros y utiliza valores por defecto si no se proporcionan.
- Si no hay datos para el estado/año seleccionados, se retorna un mensaje informativo al frontend.

4.6 Visualización y Experiencia de Usuario en el HTML

4.6.1 Interfaz de Usuario Mejorada

- Inputs interactivos: Select para estados (con todos los códigos de EE.UU.) y campo para año.
- Mensajes informativos: Alertas Bootstrap para estados de carga, éxito, advertencia y error.
- Tablas organizadas: Resultados de salario, ofertas y demanda anual presentados en tablas limpias y responsivas.
- Modal Bootstrap: Permite ver los datos crudos de la API en cada sección para transparencia y aprendizaje.
- Gráficos avanzados:
- Salario: Gráfico tipo doughnut con colores sólidos y animaciones suaves.
- Ofertas: Gráfico de barras con bordes redondeados y colores atractivos.

- Demanda anual: Gráfico de línea con área sombreada y puntos destacados.
- Contenedor principal centrado: Márgenes izquierdo y derecho, efecto glassmorphism, diseño responsive.

4.6.2 Manejo de Casos Sin Datos

- Si no hay datos para el estado/año seleccionados, se muestra un mensaje claro y no se renderizan los gráficos ni las tablas.

4.6.3 Código relevante del frontend (scripcap8.js)

```
function recargarSimulacion() {

    anioActual = document.getElementById('input-anio').value;

    estadoActual =
document.getElementById('input-estado').value;

    if (!estadoActual) {

        mostrarMensaje('Por favor selecciona un estado',
'warning');

        return;

    }

    mostrarMensaje('Cargando datos...', 'info');

    cargarSimulacion();

}

function cargarSimulacion() {


fetch(`/api/cap8/simulacion?anio=${anioActual}&estado=${estadoAc
tual}`)

        .then(resp => resp.json())
```

```
        .then(data => {
            if (data.valorProduccion === 0) {
                mostrarMensaje(`No se encontraron datos de
producción de maíz para ${estadoActual} en ${anioActual}.`,
'warning');
                return;
            }
            // ... renderizar tablas, gráficos y modales ...
        })
        .catch(err => {
            mostrarMensaje('Error al cargar los datos.',
'danger');
        });
}
function recargarSimulacion() {
    anioActual = document.getElementById('input-anio').value;
    estadoActual =
document.getElementById('input-estado').value;
    if (!estadoActual) {
        mostrarMensaje('Por favor selecciona un estado',
'warning');
        return;
    }
```

```
    mostrarMensaje('Cargando datos...', 'info');
    cargarSimulacion();
}
function cargarSimulacion() {

fetch(`/api/cap8/simulacion?anio=${anioActual}&estado=${estadoActual}`)
        .then(resp => resp.json())
        .then(data => {
            if (data.valorProduccion === 0) {
                mostrarMensaje(`No se encontraron datos de producción de maíz para ${estadoActual} en ${anioActual}.`, 'warning');
                return;
            }
            // ... renderizar tablas, gráficos y modales ...
        })
        .catch(err => {
            mostrarMensaje('Error al cargar los datos.', 'danger');
        });
}
```

8.7 Resultados y Usabilidad

- El usuario puede experimentar con diferentes estados y años, viendo el impacto en salarios y mercado laboral.
- La visualización es clara, moderna y responsiva, con feedback inmediato y acceso a los datos crudos.
- El sistema es robusto ante errores y casos sin datos, informando siempre al usuario.

5. Desarrollar la página HTML en templatate/cap8_simulacion.html

5.1 fragmento de código relevante :

```
<div class="container-main">

        <h1>Simulación Interactiva - Capítulo 8: De los salarios
del trabajo</h1>
```

```
        <!-- Inputs para experimentar con año y estado -->
        <div class="mb-4">
            <label for="input-anio"
class="form-label">Año:</label>
            <input type="number" id="input-anio"
class="form-control d-inline-block w-auto" value="2023"
min="2000" max="2023">
            <label for="input-estado" class="form-label
ms-3">Estado:</label>
            <select id="input-estado" class="form-select
d-inline-block w-auto ms-2" style="width: 200px;">
                <option value="">Seleccionar estado...</option>
                <option value="AL">AL - Alabama</option>
                <option value="AK">AK - Alaska</option>
                <option value="AZ">AZ - Arizona</option>
                <option value="AR">AR - Arkansas</option>
                <option value="CA">CA - California</option>
                <option value="CO">CO - Colorado</option>
                <option value="CT">CT - Connecticut</option>
                <option value="DE">DE - Delaware</option>
                <option value="FL">FL - Florida</option>
                <option value="GA">GA - Georgia</option>
```

```
<option value="HI">HI - Hawaii</option>
<option value="ID">ID - Idaho</option>
<option value="IL">IL - Illinois</option>
<option value="IN">IN - Indiana</option>
<option value="IA" selected>IA - Iowa</option>
<option value="KS">KS - Kansas</option>
<option value="KY">KY - Kentucky</option>
<option value="LA">LA - Louisiana</option>
<option value="ME">ME - Maine</option>
<option value="MD">MD - Maryland</option>
<option value="MA">MA - Massachusetts</option>
<option value="MI">MI - Michigan</option>
<option value="MN">MN - Minnesota</option>
<option value="MS">MS - Mississippi</option>
<option value="MO">MO - Missouri</option>
<option value="MT">MT - Montana</option>
<option value="NE">NE - Nebraska</option>
<option value="NV">NV - Nevada</option>
<option value="NH">NH - New Hampshire</option>
<option value="NJ">NJ - New Jersey</option>
<option value="NM">NM - New Mexico</option>
```

```
            <option value="NY">NY - New York</option>
            <option value="NC">NC - North Carolina</option>
            <option value="ND">ND - North Dakota</option>
            <option value="OH">OH - Ohio</option>
            <option value="OK">OK - Oklahoma</option>
            <option value="OR">OR - Oregon</option>
            <option value="PA">PA - Pennsylvania</option>
            <option value="RI">RI - Rhode Island</option>
            <option value="SC">SC - South Carolina</option>
            <option value="SD">SD - South Dakota</option>
            <option value="TN">TN - Tennessee</option>
            <option value="TX">TX - Texas</option>
            <option value="UT">UT - Utah</option>
            <option value="VT">VT - Vermont</option>
            <option value="VA">VA - Virginia</option>
            <option value="WA">WA - Washington</option>
            <option value="WV">WV - West Virginia</option>
            <option value="WI">WI - Wisconsin</option>
            <option value="WY">WY - Wyoming</option>
        </select>
```

```
            <button class="btn btn-primary ms-3"
onclick="recargarSimulacion()">Actualizar</button>
        </div>

        <!-- Mensaje de estado -->
        <div id="mensaje-estado" class="alert alert-info"
style="display: none;"></div>

        <!-- Sección 1: Salario ajustado por valor de producción
agrícola -->
        <section id="salario-produccion">
            <h2>1. Salario ajustado por valor de producción
agrícola</h2>
            <p>
                En esta sección, el salario base de un
agricultor se ajusta automáticamente según el valor real de la
producción de maíz en el estado y año seleccionados, usando
datos de la API de USDA. Esto ilustra cómo la riqueza generada
en el sector puede influir en los salarios del trabajo, como
describe Adam Smith.
            </p>
            <table class="sim-data" style="width:auto;">
                <tr><th>Salario base ajustado</th><th>Valor de
producción</th><th>Ver datos de la API</th></tr>
```

```
                <tr>
                    <td id="salario-ajustado"></td>
                    <td id="valor-produccion"></td>
                    <td>
                        <button class="btn btn-secondary btn-sm"
onclick="mostrarModalApi('apiRaw1')">Ver datos de la
API</button>
                    </td>
                </tr>
            </table>
            <div id="explicacion-salario" style="margin-top:1em;
color:#555;"></div>
            <!-- Contenedor para gráfico -->
            <div style="height: 200px; position: relative;
margin-top: 20px;">
                <canvas id="grafico-salario"></canvas>
            </div>
        </section>
```

5.2 Creando style.css del capítulo 8 en assen/css/stylecap8.css

fragmento de código relevante :

```
body {
```

```
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana,
sans-serif;
    margin: 0;
    padding: 20px;
    background: linear-gradient(135deg, #667eea 0%, #764ba2
100%);
    min-height: 100vh;
    color: #333;
}


.container-main {
    max-width: 1200px;
    margin: 0 auto;
    padding: 0 20px;
    background: rgba(255, 255, 255, 0.95);
    border-radius: 15px;
    box-shadow: 0 10px 40px rgba(0,0,0,0.15);
    backdrop-filter: blur(10px);
    border: 1px solid rgba(255, 255, 255, 0.2);
}
```

```
h1 {
    color: #2c3e50;
    text-align: center;
    margin-bottom: 30px;
    font-weight: 600;
    padding-top: 20px;
}
```

...

5,3 Creación de js en assets/js/scripca8.js

fragmento de código relevante:

```
let datosSimulacion = null;
let estadoActual = 'IA';
let anioActual = 2023;


function mostrarModalApi(key) {
    if (!datosSimulacion) return;
    let contenido = '';
    if (key === 'apiRaw1') contenido =
JSON.stringify(datosSimulacion.apiRaw1, null, 2);
    if (key === 'apiRaw2') contenido =
JSON.stringify(datosSimulacion.apiRaw2, null, 2);
    if (key.startsWith('apiRawDemanda')) {
```

```
        const anio = key.split('-')[1];
        contenido =
JSON.stringify(datosSimulacion.apiRawDemanda[anio], null, 2);
    }
    document.getElementById('modalApiContent').innerText =
contenido;
    const modal = new
bootstrap.Modal(document.getElementById('modalApi'));
    modal.show();
}

function recargarSimulacion() {
    anioActual = document.getElementById('input-anio').value;
    estadoActual =
document.getElementById('input-estado').value;

    if (!estadoActual) {
        mostrarMensaje('Por favor selecciona un estado',
'warning');
        return;
    }

    mostrarMensaje('Cargando datos...', 'info');
```

```
    cargarSimulacion();

}
```

....

6.Conclusión

La simulación desarrollada para el Capítulo 8 de "La Riqueza de las Naciones" trasciende la simple visualización de datos, permitiendo experimentar de forma interactiva con los principios fundamentales de la economía laboral propuestos por Adam Smith. A través de la integración con la API de USDA y una interfaz moderna, el usuario puede observar y analizar:

- 1. Salario ajustado por valor de producción agrícola:

Se simula cómo el salario de los trabajadores agrícolas varía en función de la riqueza generada por la producción real de maíz en cada estado y año, reflejando el concepto de salario de mercado influido por la oferta y la demanda.

- 2. Número de ofertas laborales según producción:

Se modela la relación directa entre el volumen de producción agrícola y la cantidad de ofertas laborales disponibles, ilustrando cómo la actividad económica y la demanda de trabajo afectan las oportunidades de empleo.

- 3. Demanda laboral ajustada por año:

Se permite analizar la evolución de la demanda de trabajo agrícola a lo largo de varios años, mostrando tendencias y el impacto de factores externos sobre el mercado laboral.Cada uno de estos puntos emula los mecanismos descritos por Adam Smith: el salario natural y de mercado, la influencia de la producción y la economía en los salarios, y la dinámica de la oferta y demanda laboral. La simulación no solo facilita la comprensión teórica, sino que la lleva a la práctica con datos reales y visualizaciones interactivas, convirtiéndose en una herramienta educativa y analítica de gran valor.