

## **Activity # 1**

### **Client Server Web Application**

Our team is Team 4 and is called “RIOT” which is a recursive acronym that stands for “RIOT Is Our Team”, like YAML or GNU. Our team consists of:

- Sanguña Lanchimba Robert Denilson
- Yáñez Tufiño Mishell Alexandra
- Yáñez Freire José Ignacio
- Troya Acosta Alexis Santiago

We'll explain our work related to each of the items considered in the Rubric

#### **1. GitHub Repository**

Our Project can be accessed here:

[https://github.com/joseignacioyanez/Team4\\_RIOT\\_InvoicingSystem](https://github.com/joseignacioyanez/Team4_RIOT_InvoicingSystem)

We followed the structure of files which allowed us to develop the software from Definition until Deployment in the Cloud.

#### **2. Idea (interview) and list of features (product Backlog)**

The interview is a very useful tool to know the needs of the client.

List of features.

- Manage users.
- Manage products.
- Generate invoice.
- Register payment method.

**Interview with the restaurant owner.**



## Photo of the restaurant



We needed to understand the process of Scrum, so we read the first chapters of a book called “Head First Agile: A Brain-Friendly Guide to Agile Principles, Ideas, and Real-World Practices”. Here we understood how other teams organize their work based on User Stories and we made them based on this type of cards:



### User Story 1

As the Restaurant Owner

I want to be able to create personal and secure users for the cashiers

So that they can register sales and give Invoices to the clients

### User Story 2

As the Restaurant Owner

I want to be able to manage the restaurant menu

So that Invoices have the details of items that were bought

### User Story 3

As the Cashier

I want to be able to create and search clients by their ID or RUC

So that I can associate them to the Invoice

### User Story 4

As the Cashier

I want to be able to generate a detailed Invoice and send it by Email

So that the Client has the Invoice

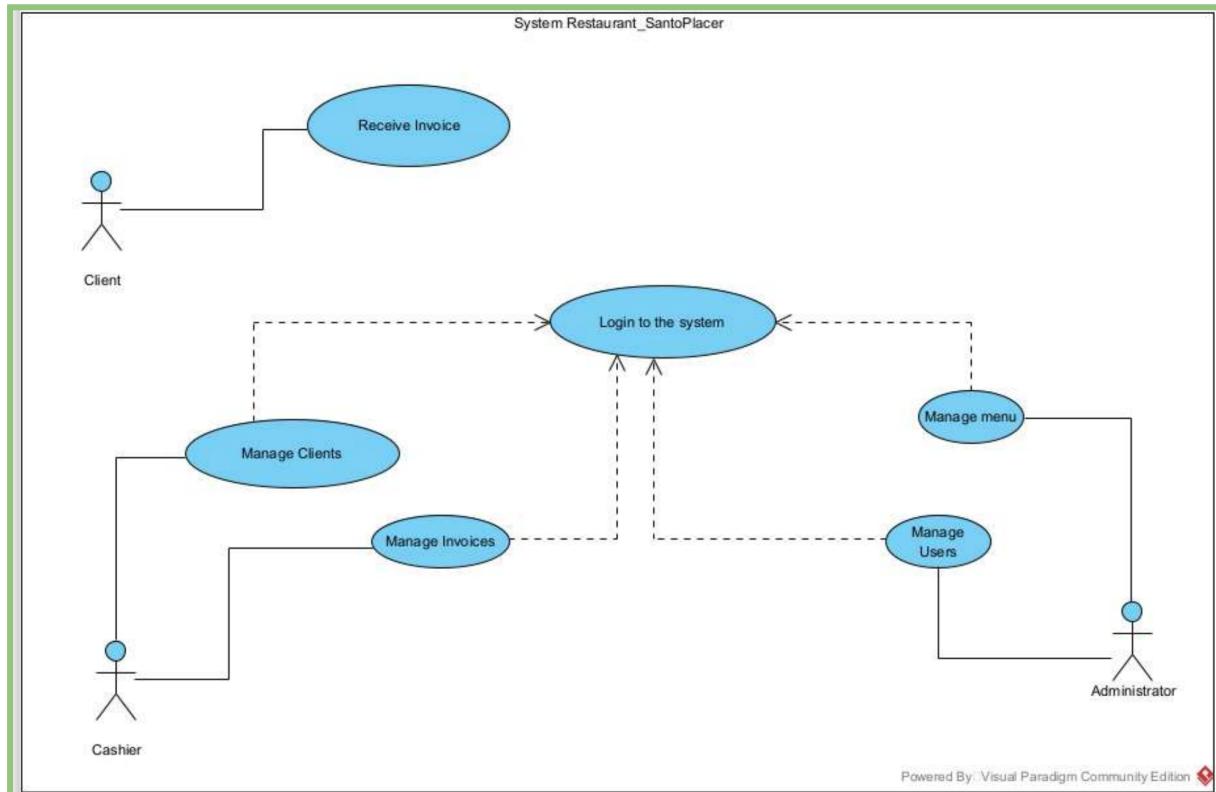
Then we divided the work into tasks based on the User Stories that were assigned to each one of us and started registering this on a Kanban Board using Trello.

The screenshot shows a Trello board titled "Scrum Actividad 1". The board is divided into four main sections: Backlog, Sprint Backlog, Desarrollo, and Sprint Finalizado. The Sprint Backlog section contains a single card: "Sprint Meta: Entregar Aplicacion con 4 Requerimientos Funcionales". The Desarrollo section contains two cards: "Vista de Inicio de Sesión" and "Vista de Requisito 2". The Sprint Finalizado section contains three cards: "1. Definición del Proyecto", "Historia de Usuario 2 Tarjeta", and "Vistas del Yanez". Each card has a small circular badge in the bottom right corner with initials: JF, A, MT, or RL.

### 3. Diagrams (Classes, Use Cases, Architecture)

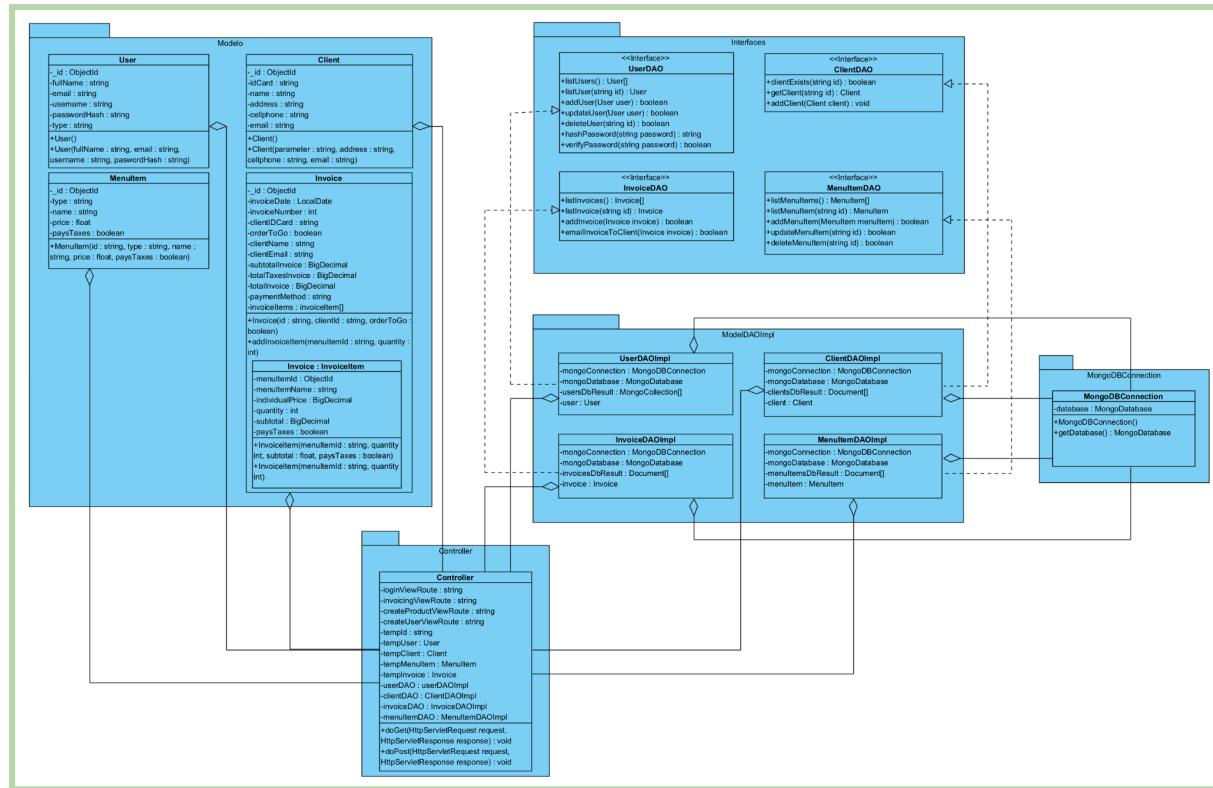
Through use cases we can describe the steps or activities to be carried out in each process.

**Use Cases Diagram**

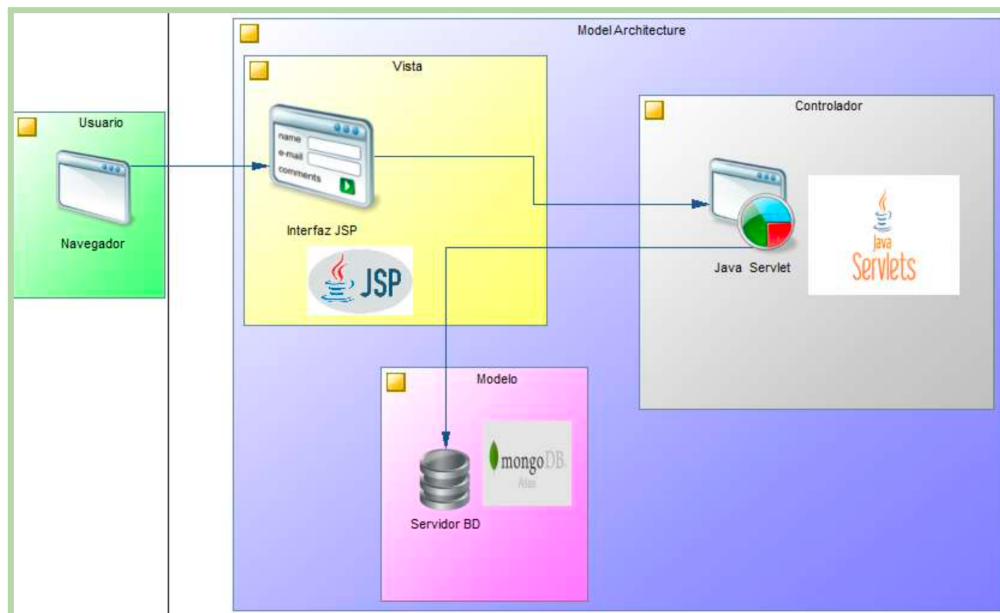


## Class Diagram

The classes that make up the Web Application are described below, as well as its methods and attributes.



And this is the general Architecture we followed:



## 4. MongoDB Atlas Database (in the cloud)

We created the Mongo Database using Atlas. It has 4 Collections for the 4 Classes or web app uses.

The screenshot shows the MongoDB Atlas interface. On the left, there's a sidebar with 'DATABASES: 1' and 'COLLECTIONS: 4'. Below this are buttons for '+ Create Database' and a search bar 'Search Namespaces'. A tree view shows a database named 'WAD\_1\_Invoicing\_SantoPlacer' containing collections: 'clients', 'invoices', 'menuitems' (which is selected), and 'users'. The main panel is titled 'WAD\_1\_Invoicing\_SantoPlacer.menuItems'. It displays storage size (4KB), logical data size (0B), total documents (0), and index size (0). There are tabs for 'Find', 'Indexes', 'Schema Anti-Patterns', and 'Aggregations'. A 'FILTER' button with the query '{ field: 'value' }' is present. The results section shows 'QUERY RESULTS: 1-6 OF 6' with two documents:

```
_id: ObjectId('637ee5eb74565d726b2ba62b')
type: "alimentos"
name: "Sopa Grande"
price: 2.34
paysTaxes: true

_id: ObjectId('637ee61474565d726b2c033e')
type: "alimentos"
name: "Plato Segundo"
price: 3.1
paysTaxes: true
```

While researching about MongoDB, we read about how it allows us to have the data we use at the same time, all in one place. So that's why we made the Class Invoice and it includes fields for details of the Client and the MenuItems. First, we don't have to make many queries, just one to generate all the Invoice. And second, Invoices shouldn't be able to update or change, just archived, so this duplication of data is not that important.

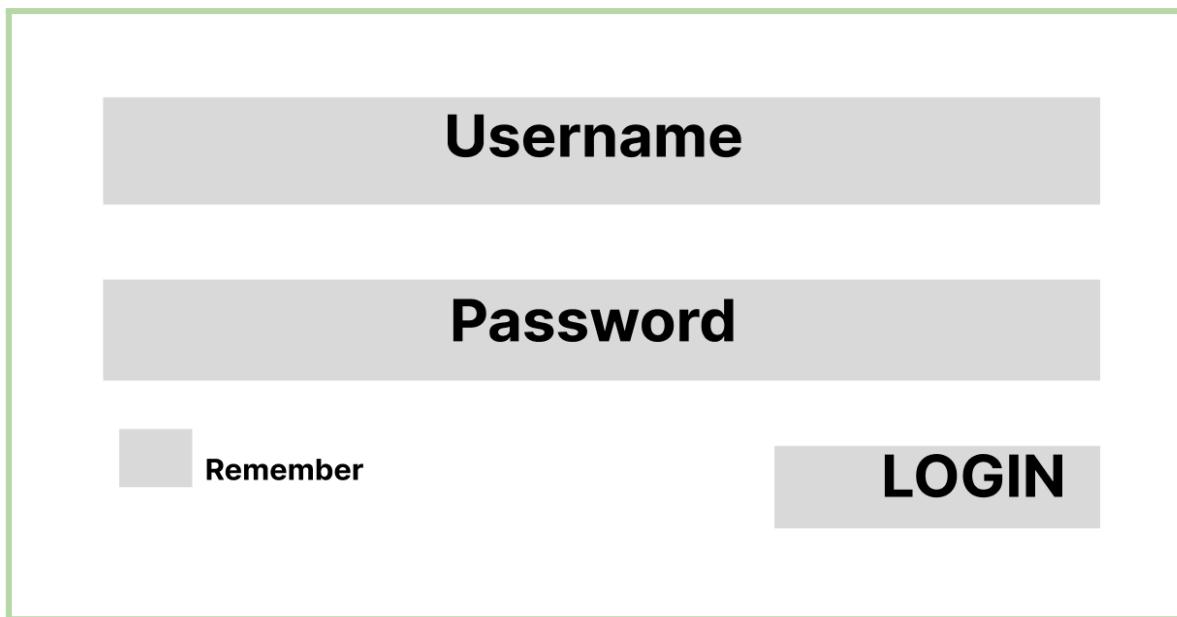
```
_id: ObjectId('637ee79574565d726b2f7103')
clientIDCard: "0501751316"
orderToGo: false
clientName: "Felipe Fernandez"
clientEmail: "felixoxo@hotmail.com"
subtotalInvoice: 5.2
totalTaxesInvoice: 0.62
totalInvoice: 5.82
invoiceItems: Array
  0: Object
    menuItemId: ObjectId('637da261cbaa31c9c23359e1')
    menuItemName: "Limonada Grande"
    individualPrice: 1.1
    quantity: 2
    subtotal: 2.2
    paysTaxes: true
  1: Object
    menuItemId: ObjectId('637da2f3cbaa31c9c23359e2')
    menuItemName: "Postre de Guayaba"
    individualPrice: 3
    quantity: 1
    subtotal: 3
    paysTaxes: true
    paymentMethod: "TarjetaCredito"
    invoiceNumber: 1
```

We also made a nested class InvoiceItem so we can have the details for each “row” of the invoice and even have products that pay taxes or not, and all being counted.

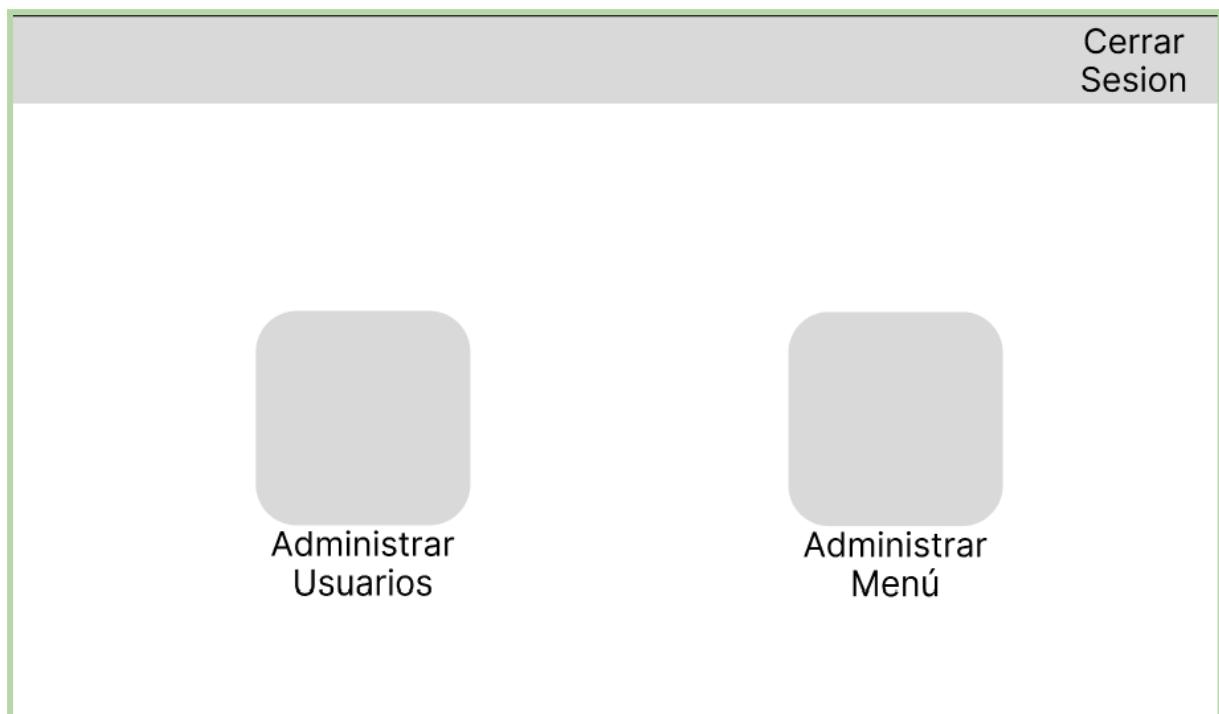
## 5. Design of the client and at least four business rules

For the structure of the views of the Web Application, a sketch was made of each one on Figma.

**Login view**



**Home view**



**Menu view**

				Crear
Código	Nombre	Editar	Estado	
001	Almuerzo	Editar	Activo	

**Users view**

					Inicio
Cedula	Nombre	Usuario	Editar	Estado	
1753506128	Mishell Yáñez	Mishu	Editar	Activo	

**Product creation view**

Cancelar

**Crear Producto**

Activo

Código

Categoría Alimentos /  
logistica

Nombre

Precio

IVA

**Guardar**

Cancelar

**Modificar Producto**

Activo

Código

Categoría Alimentos /  
logistica

Nombre

Precio

IVA

**Confirmar**

Cancelar

### Crear Nuevo Usuario

Cédula:

Nombre Completo:

Usuario:

Contraseña:

Repetir Contraseña:

Confirmar

Cancelar

### Modificar Usuario

Cédula:

Nombre Completo:

Usuario:

Contraseña:

Repetir Contraseña:

Confirmar

And finally we designed the HTML and Styles of the Views using Bootstrap.



### Inicio de Sesión

Usuario o Contraseña Incorrectos

Usuario  
jiyf

Contraseña

Iniciar Sesión

RIOT [RIOT Is Our Team] - WAD - 2022



### Menú del Administrador

Cerrar Sesión

Administrador Usuarios

Administrador Menú

Usuarios				
		Crear Usuario		
Nombre	Email	Usuario	Editar	Eliminar
Jose Ignacio Yanez	jiyanez1@espe.edu.ec	jiyf	<button>Editar</button>	<button>Eliminar</button>
Mishell Alexandra Yanez	mayanez2@espe.edu.ec	mayt	<button>Editar</button>	<button>Eliminar</button>
Fernando Patricio Fernandez	ffernandez@hotmail.com	ffernandez	<button>Editar</button>	<button>Eliminar</button>
Robert Denilson Sanguña	rdsanguna4@espe.edu.ec	rDSL	<button>Editar</button>	<button>Eliminar</button>
Patricia Felicia Morticia	paty@pami.com	pppp	<button>Editar</button>	<button>Eliminar</button>

### Business rules

- Password encryption.
- Send invoices by mail.
- Sales report.
- Report of invoices pending payment.
- Generate prepaid invoices.

### 6. Execution of the application in the cloud or hosting

For deployment we didn't have too much experience so we spent many hours researching, solving problems with environment variables of Maven and Java, installing plugins for Java packaging, trying to understand the purpose and syntax of app.yaml and appengine-web.xml, deciding if using AppEngine or a VM of Compute Engine and run manually the app, and so on.

```
[INFO] --- appengine-maven-plugin:2.4.4:deploy (default-cli) @ WAD RIOT_ClientServer_InvoicingSystem ---
[INFO] Staging the application to: /Users/joseignacio/NetBeansProjects/WAD RIOT_ClientServer_InvoicingSystem/target/appengine-staging
[INFO] Detected App Engine app.yaml based application.
[INFO] -----
[INFO] BUILD FAILURE
[INFO] -----
[INFO] Total time:  4.184 s
[INFO] Finished at: 2022-11-23T19:32:08-05:00
[INFO] -----
[ERROR] Failed to execute goal com.google.cloud.tools:appengine-maven-plugin:2.4.4:deploy (default-cli) on project WAD RIOT_ClientServer_InvoicingSystem: Execution default-cli of goal com.google.cloud.tools:appengine-maven-plugin:2.4.4:deploy failed: com.google.cloud.tools.appengine.AppEngineException: java.nio.file.NoSuchFileException: /Users/joseignacio/NetBeansProjects/WAD RIOT_ClientServer_InvoicingSystem/src/main/appengine/app.yaml -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
```

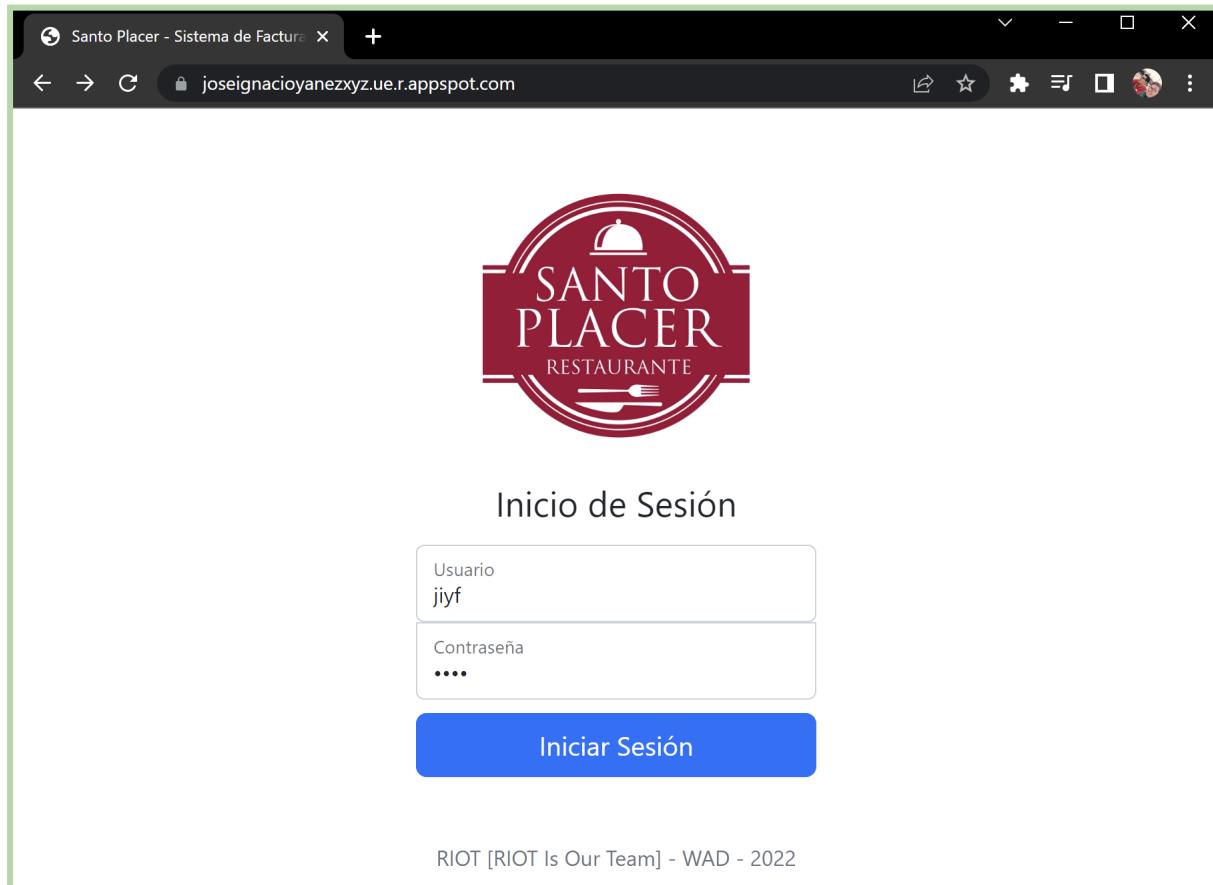
```
:C 1 WAD RIOT_ClientServer_InvoicingSystem $ gcloud app deploy
Services to deploy:

descriptor:           [/Users/joseignacio/NetBeansProjects/WAD RIOT_ClientServer_InvoicingSystem/pom.xml]
source:              [/Users/joseignacio/NetBeansProjects/WAD RIOT_ClientServer_InvoicingSystem]
target project:      [joseignacioyerxyz]
target service:      [default]
target version:      [20221123t200529]
target url:          [https://joseignacioyerxyz.ue.r.appspot.com]
target service account: [App Engine default service account]

Do you want to continue (Y/n)? y
Beginning deployment of service [default]...
Uploading 5 files to Google Cloud Storage
File upload done.
Updating service [default]...failed.
ERROR: (gcloud.app.deploy) Error Response: [9] Cloud build 19568d26-3bb9-4b1a-bd0d-4707fbdf5a18 status: FAILURE
did not find any jar files with a Main-Class manifest entry
Full build logs: https://console.cloud.google.com/cloud-build/builds;region=us-east1/19568d26-3bb9-4b1a-bd0d-4707fb
:( 1 WAD RIOT_ClientServer_InvoicingSystem $ mvn package appengine:deploy -Dapp.deploy.projectId=joseignacioyerxyz
```

After too many errors, one time, the app finally deployed to Google Cloud AppEngine and it can be accessed here:

<https://joseignacioyerxyz.ue.r.appspot.com/>



The thing is that it does not have all the functionality of the app running on a local machine. It has some errors that we couldn't easily debug because of it being in the cloud, and that the time was running out. We hope to be able to fix this.

## **Video of the Web App Functioning and Deployed in the Web**

<https://youtu.be/XrfRwmEvP7A>

### **Some References we read when trying to deploy to the Cloud**

- <https://cloud.google.com/appengine/docs/standard/testing-and-deploying-your-app?tab=ab=java>
- <https://github.com/GoogleCloudPlatform/app-maven-plugin>
- <https://www.baeldung.com/executable-jar-with-maven>
- <https://stackoverflow.com/questions/55065465/using-google-app-engine-with-java-gives-url-was-not-found-on-this-server-but-w>
- <https://medium.com/oracledevs/run-java-ee-8-apps-on-oracle-cloud-with-payara-micro-d9b527adaac9>
- <https://codelabs.developers.google.com/codelabs/cloud-app-engine-springboot#7>
- <https://cloud.google.com/appengine/docs/legacy/standard/java/using-maven>
-  Deploying a Java Application to Google App Engine (GCP) | Complete Tutorial ...
-  Deploying Java Web App to App Engine Standard