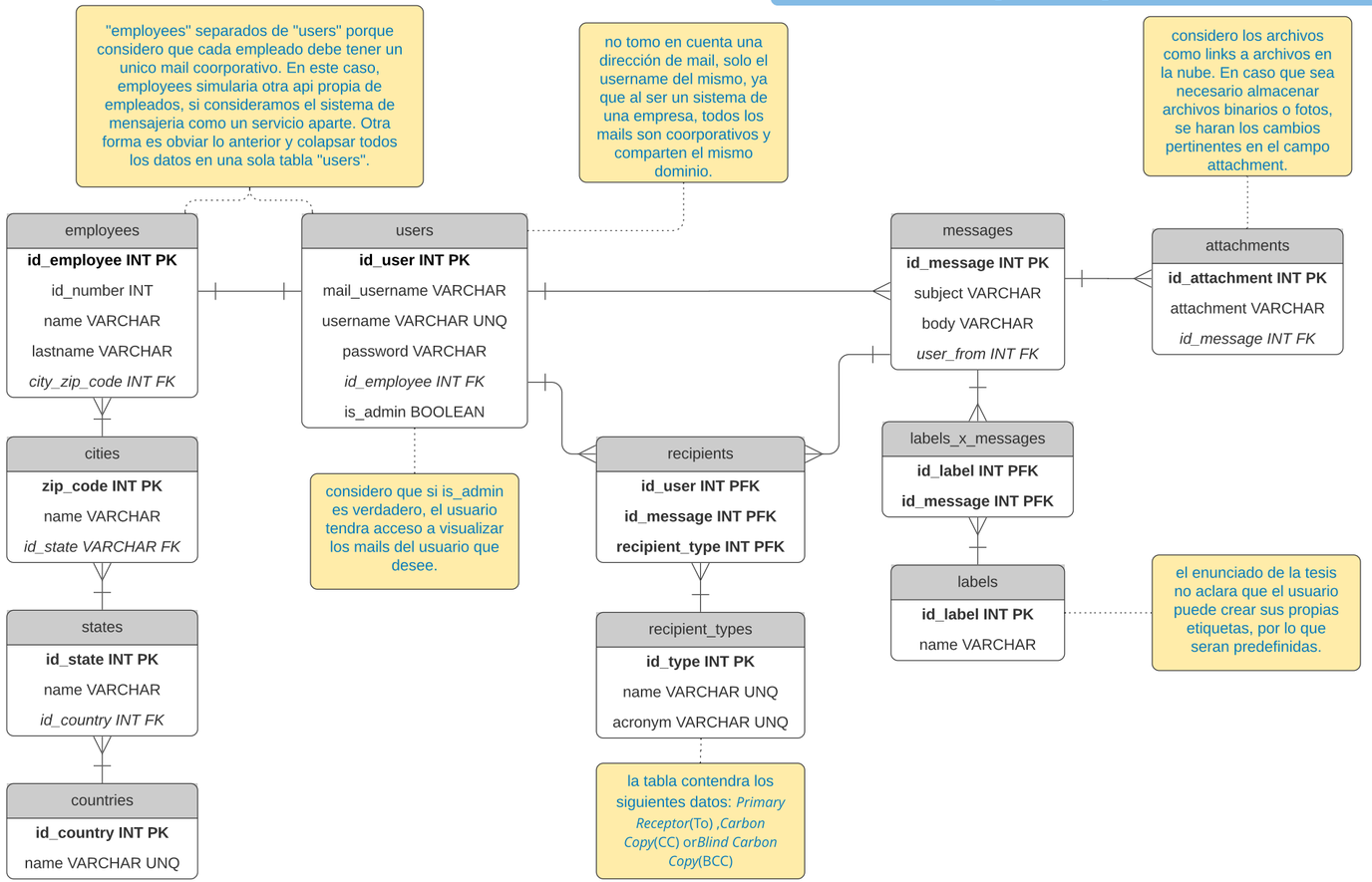
ERD (UML) - Final Thesis Work - V1



Final Thesis work

This is your final thesis to be a certified Glober in Java Web Services.

We need to create a Web Service to allow a company handle a internal messages between employees. Our web service should give support to the following functional requirements :

1. **User registration** : We should have the following data : username , name, last name, identification number, address, zip code city, state, country
2. **User login** : Web service have to have the ability to login a registerd user to send and receive messages.
3. **Message sending** : Web service should allow to send messages to multiple registered users specifiying each recipient as a *Primary Receptor* (To) , *Carbon Copy* (CC) or *Blind Carbon Copy* (BCC). User should be logged to sendand receive messages messages. Messages should have the following data : *Subject*, *Body* and *Attachments*.
4. **Message Reception** : Web service should allow to receive messages . The user can receive message and catalogue each one with one or more labels.
5. **Inbox and Sent** : Web service will provide support to query messages in inbox and sent.
6. **Label Filtering** : Web service must provide tools to filter messages by label. User can add different labels to each message. Each user can create their own personalized Web Services.

Our Web Service should meet the following Non-Functional requirements :

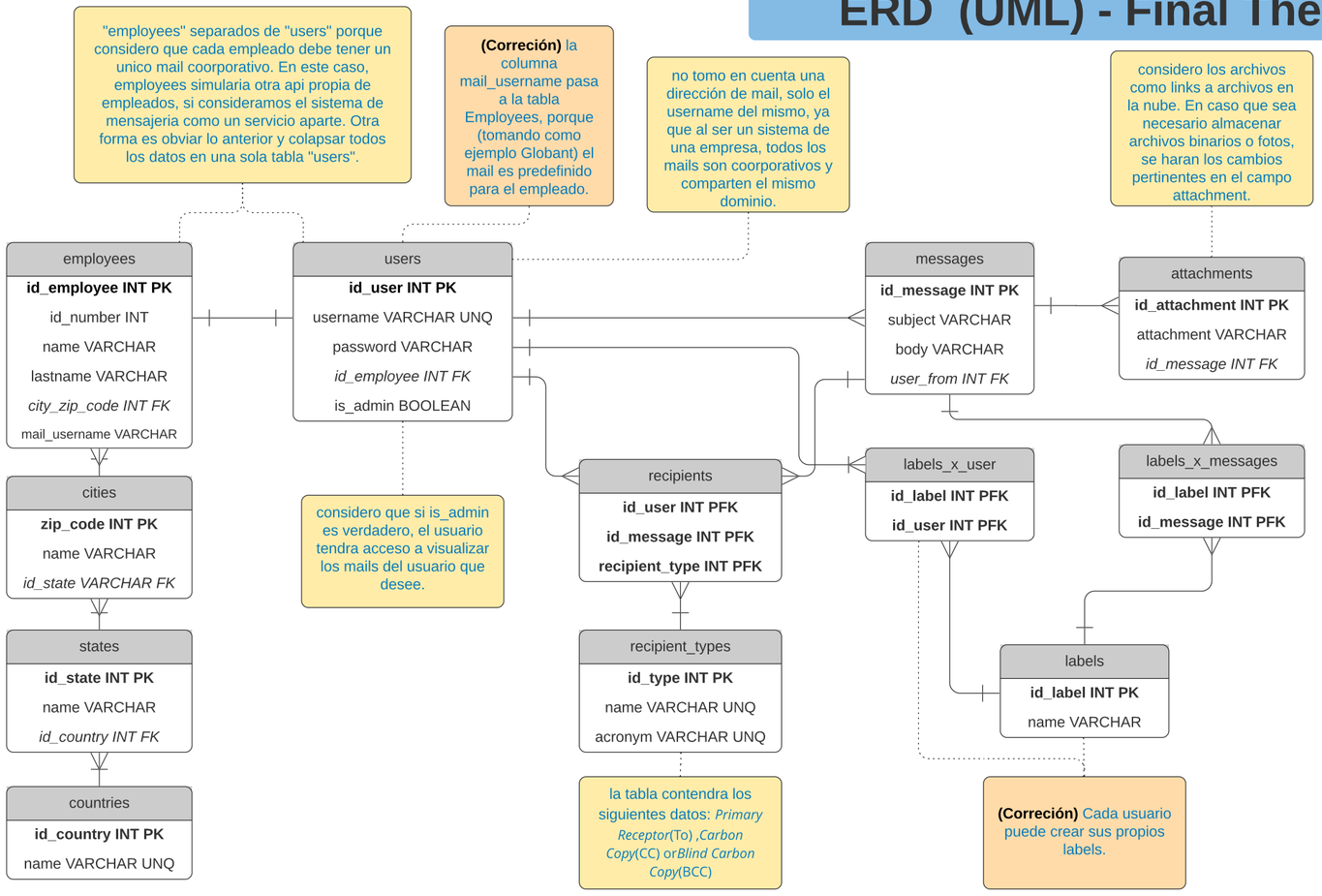
1. **RESTFUL** : Web service should comply the REST principles.
2. **Pagination** : Web service must allow pagination for queries.
3. **Performance** : Web service must respond each in less than 10 milliseconds
4. **Use design patterns and good practices** : You must use Design patterns learned in this course.
5. **Testing** : You must reach at 80% of Unit Testing coverage in your solution.

You can use the storage you consider the best to provide support to solve this problem. We prefer to use a Relational Database , but if you consider if a NoSQL database fits to comply requirements it's ok.

Optional Activity

In order to have a more comprehensive picture on how Web Services works in a *normal* environment, you can Dockerize the solution creating containers in an internal network for DB and Web Service and connect both.

ERD (UML) - Final Thesis Work - V2



Final Thesis work

This is your final thesis to be a certified Glober in Java Web Services.

We need to create a Web Service to allow a company handle a internal messages between employees. Our web service should give support to the following functional requirements :

1. **User registration** : We should have the following data : username , name, last name, identification number, address, zip code city, state, country
2. **User login** : Web service have to have the ability to login a registerd user to send and receive messages.
3. **Message sending** : Web service should allow to send messages to multiple registered users specifiying each recipient as a *Primary Receptor* (To) , *Carbon Copy* (CC) or *Blind Carbon Copy* (BCC). User should be logged to sendand receive messages messages. Messages should have the following data : *Subject*, *Body* and *Attachments*.
4. **Message Reception** : Web service should allow to receive messages . The user can receive message and catalogue each one with one or more labels.
5. **Inbox and Sent** : Web service will provide support to query messages in inbox and sent.
6. **Label Filtering** : Web service must provide tools to filter messages by label. User can add different labels to each message. Each user can create their own personalized Web Services.

Our Web Service should meet the following Non-Functional requirements :

1. **RESTFUL** : Web service should comply the REST principles.
2. **Pagination** : Web service must allow pagination for queries.
3. **Performance** : Web service must respond each in less than 10 milliseconds
4. **Use design patterns and good practices** : You must use Design patterns learned in this course.
5. **Testing** : You must reach at 80% of Unit Testing coverage in your solution.

You can use the storage you consider the best to provide support to solve this problem. We prefer to use a Relational Database , but if you consider if a NoSQL database fits to comply requirements it's ok.

Optional Activity

In order to have a more comprehensive picture on how Web Services works in a *normal* environment, you can Dockerize the solution creating containers in an internal network for DB and Web Service and connect both.

Comentario de Nacho a la V1:

- Labels: El usuario puede crear sus propios labels. Solo que hay un typo en el enunciado del ejercicio (Igual si son predefinidos, no va a estar mal tampoco).

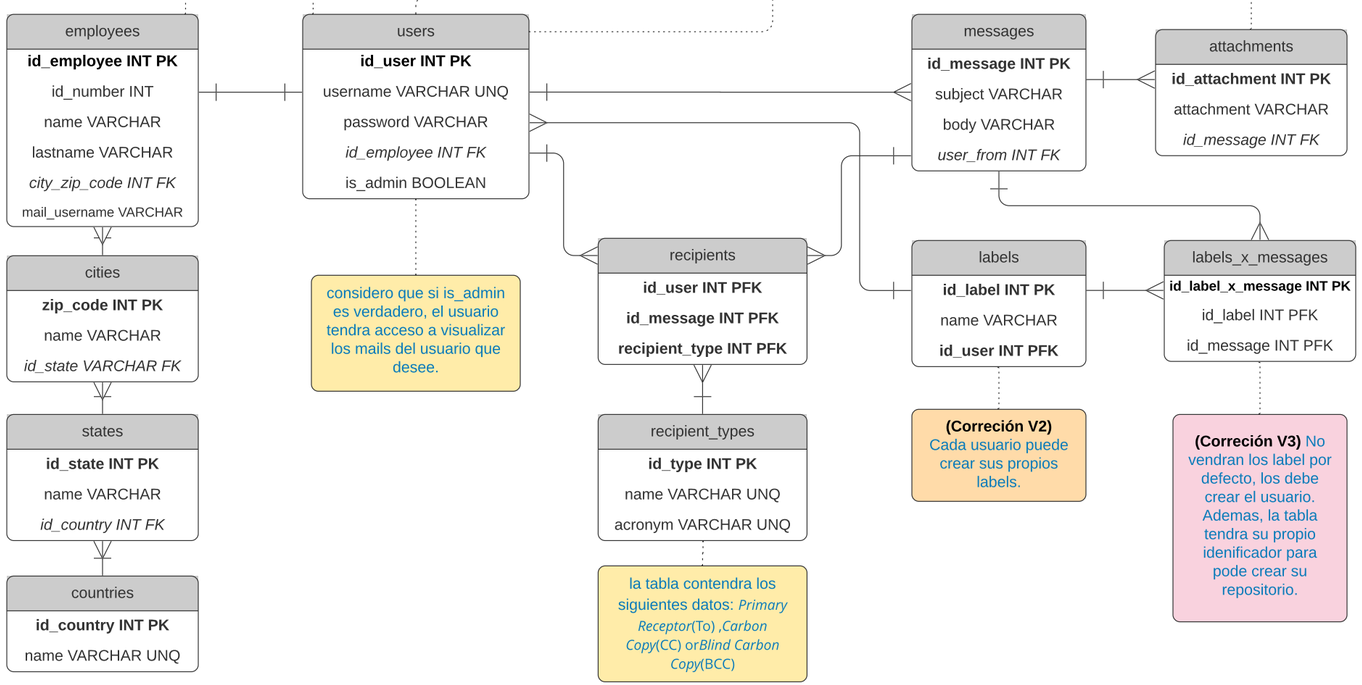
ERD (UML) - Final Thesis Work - V1

"employees" separados de "users" porque considero que cada empleado debe tener un unico mail coorporativo. En este caso, employees simularia otra api propia de empleados, si consideramos el sistema de mensajeria como un servicio aparte. Otra forma es obviar lo anterior y colapsar todos los datos en una sola tabla "users".

(Correción V2) la columna mail_username pasa a la tabla Employees, porque (tomando como ejemplo Globant) el mail es predefinido para el empleado.

no tomo en cuenta una dirección de mail, solo el username del mismo, ya que al ser un sistema de una empresa, todos los mails son coorporativos y comparten el mismo dominio.

considero los archivos como links a archivos en la nube. En caso que sea necesario almacenar archivos binarios o fotos, se haran los cambios pertinentes en el campo attachment.



considero que si is_admin es verdadero, el usuario tendra acceso a visualizar los mails del usuario que desee.

la tabla contendra los siguientes datos: *Primary Receptor*(To) ,*Carbon Copy*(CC) or*Blind Carbon Copy*(BCC)

(Correción V2) Cada usuario puede crear sus propios labels.

(Correción V3) No vendran los label por defecto, los debe crear el usuario. Ademas, la tabla tendra su propio idenificador para pode crear su repositorio.

Final Thesis work

This is your final thesis to be a certified Glober in Java Web Services.

We need to create a Web Service to allow a company handle a internal messages between employees. Our web service should give support to the following functional requirements :

1. **User registration** : We should have the following data : username , name, last name, identification number, address, zip code city, state, country
2. **User login** : Web service have to have the ability to login a registerd user to send and receive messages.
3. **Message sending** : Web service should allow to send messages to multiple registered users specifiying each recipient as a *Primary Receptor* (To) , *Carbon Copy* (CC) or *Blind Carbon Copy* (BCC). User should be logged to sendand receive messages messages. Messages should have the following data : *Subject*, *Body* and *Attachments*.
4. **Message Reception** : Web service should allow to receive messages . The user can receive message and catalogue each one with one or more labels.
5. **Inbox and Sent** : Web service will provide support to query messages in inbox and sent.
6. **Label Filtering** : Web service must provide tools to filter messages by label. User can add different labels to each message. Each user can create their own personalized Web Services.

Our Web Service should meet the following Non-Functional requirements :

1. **RESTFUL** : Web service should comply the REST principles.
2. **Pagination** : Web service must allow pagination for queries.
3. **Performance** : Web service must respond each in less than 10 milliseconds
4. **Use design patterns and good practices** : You must use Design patterns learned in this course.
5. **Testing** : You must reach at 80% of Unit Testing coverage in your solution.

You can use the storage you consider the best to provide support to solve this problem. We prefer to use a Relational Database , but if you consider if a NoSQL database fits to comply requirements it's ok.

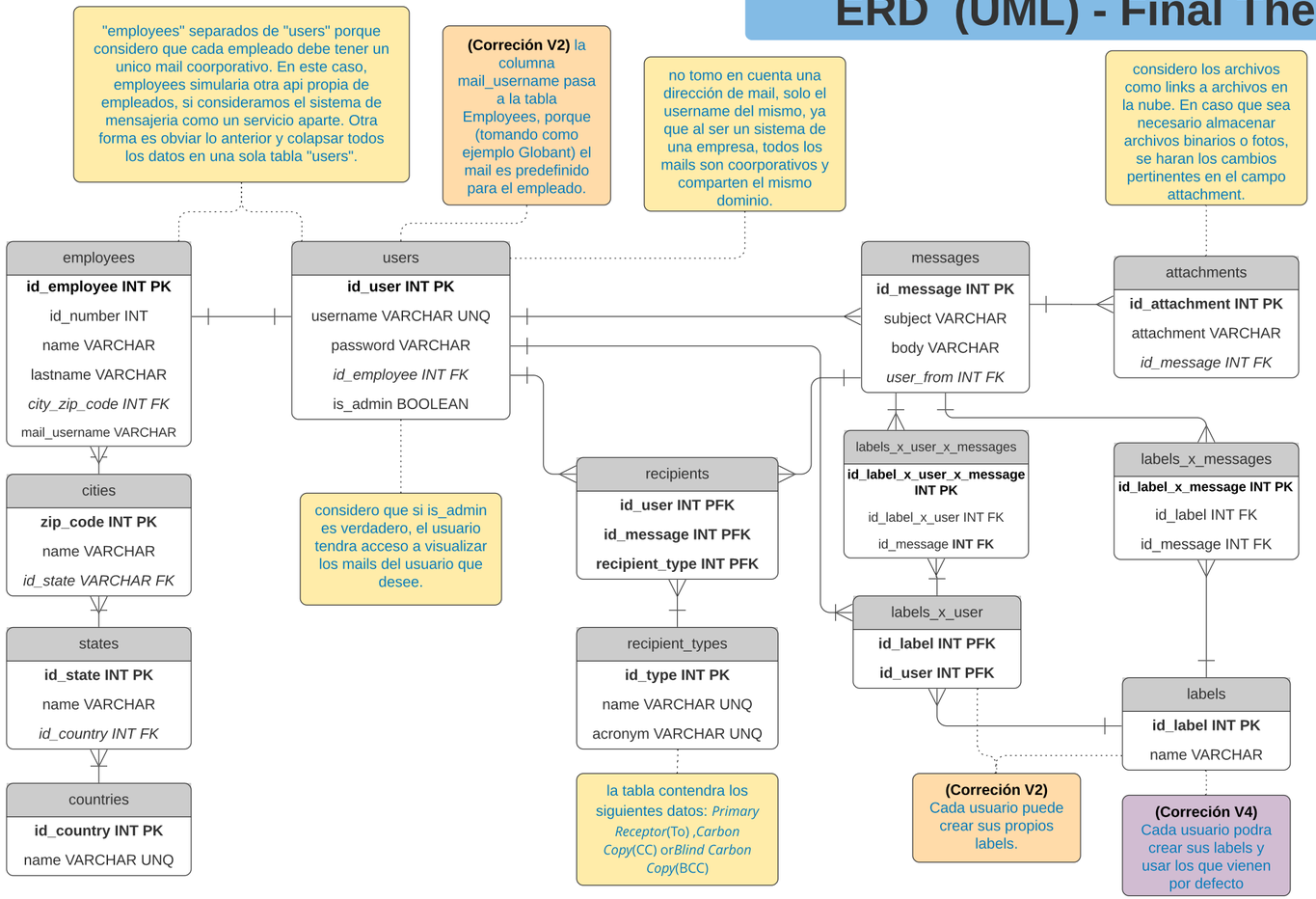
Optional Activity

In order to have a more comprehensive picture on how Web Services works in a *normal* environment, you can Dockerize the solution creating containers in an internal network for DB and Web Service and connect both.

Comentario de Nacho a la V1 (Correciones en V2):

- Labels: El usuario puede crear sus propios labels. Solo que hay un typo en el enunciado del ejercicio (Igual si son predefinidos, no va a estar mal tampoco).

ERD (UML) - Final Thesis Work - V4



Final Thesis work

This is your final thesis to be a certified Glober in Java Web Services.

We need to create a Web Service to allow a company handle a internal messages between employees. Our web service should give support to the following functional requirements :

1. **User registration** : We should have the following data : username , name, last name, identification number, address, zip code city, state, country
2. **User login** : Web service have to have the ability to login a registerd user to send and receive messages.
3. **Message sending** : Web service should allow to send messages to multiple registered users specifiying each recipient as a *Primary Receptor* (To) , *Carbon Copy* (CC) or *Blind Carbon Copy* (BCC). User should be logged to sendand receive messages messages. Messages should have the following data : *Subject*, *Body* and *Attachments*.
4. **Message Reception** : Web service should allow to receive messages . The user can receive message and catalogue each one with one or more labels.
5. **Inbox and Sent** : Web service will provide support to query messages in inbox and sent.
6. **Label Filtering** : Web service must provide tools to filter messages by label. User can add different labels to each message. Each user can create their own personalized Web Services.

Our Web Service should meet the following Non-Functional requirements :

1. **RESTFUL** : Web service should comply the REST principles.
2. **Pagination** : Web service must allow pagination for queries.
3. **Performance** : Web service must respond each in less than 10 milliseconds
4. **Use design patterns and good practices** : You must use Design patterns learned in this course.
5. **Testing** : You must reach at 80% of Unit Testing coverage in your solution.

You can use the storage you consider the best to provide support to solve this problem. We prefer to use a Relational Database , but if you consider if a NoSQL database fits to comply requirements it's ok.

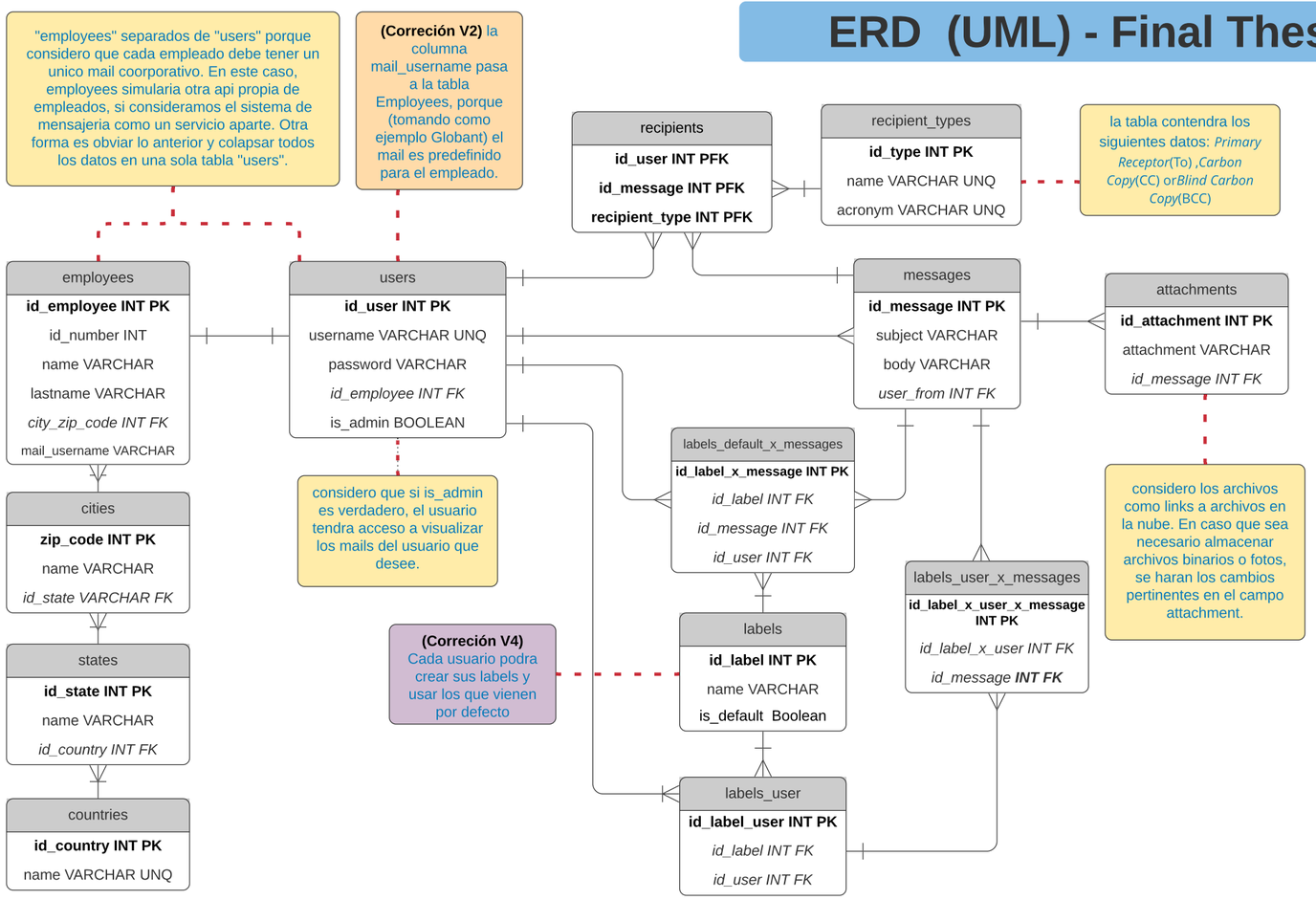
Optional Activity

In order to have a more comprehensive picture on how Web Services works in a *normal* environment, you can Dockerize the solution creating containers in an internal network for DB and Web Service and connect both.

Comentario de Nacho a la V1:

- Labels: El usuario puede crear sus propios labels. Solo que hay un typo en el enunciado del ejercicio (Igual si son predefinidos, no va a estar mal tampoco).

ERD (UML) - Final Thesis Work - V5



Final Thesis work

This is your final thesis to be a certified Glober in Java Web Services.

We need to create a Web Service to allow a company handle a internal messages between employees. Our web service should give support to the following functional requirements :

1. **User registration** : We should have the following data : username , name, last name, identification number, address, zip code city, state, country
2. **User login** : Web service have to have the ability to login a registerd user to send and receive messages.
3. **Message sending** : Web service should allow to send messages to multiple registered users specifiying each recipient as a *Primary Receptor* (To) , *Carbon Copy* (CC) or *Blind Carbon Copy* (BCC). User should be logged to sendand receive messages messages. Messages should have the following data : *Subject*, *Body* and *Attachments*.
4. **Message Reception** : Web service should allow to receive messages . The user can receive message and catalogue each one with one or more labels.
5. **Inbox and Sent** : Web service will provide support to query messages in inbox and sent.
6. **Label Filtering** : Web service must provide tools to filter messages by label. User can add different labels to each message. Each user can create their own personalized Web Services.

Our Web Service should meet the following Non-Functional requirements :

1. **RESTFUL** : Web service should comply the REST principles.
2. **Pagination** : Web service must allow pagination for queries.
3. **Performance** : Web service must respond each in less than 10 milliseconds
4. **Use design patterns and good practices** : You must use Design patterns learned in this course.
5. **Testing** : You must reach at 80% of Unit Testing coverage in your solution.

You can use the storage you consider the best to provide support to solve this problem. We prefer to use a Relational Database , but if you consider if a NoSQL database fits to comply requirements it's ok.

Optional Activity

In order to have a more comprehensive picture on how Web Services works in a *normal* environment, you can Dockerize the solution creating containers in an internal network for DB and Web Service and connect both.

Comentario de Nacho a la V1:

- Labels: El usuario puede crear sus propios labels. Solo que hay un typo en el enunciado del ejercicio (Igual si son predefinidos, no va a estar mal tampoco).