Graphical user interface, text

Description automatically generated

**Lesson 1 Demo 3**

**Create Chaincode for Cross Border Payment**

|  |
| --- |
| **Objective:** To create Chaincode for a Cross Border Payment  **Tools required:** Ubuntu and Eclipse IDE  **Prerequisites:** Lesson 1 Demo 2 |

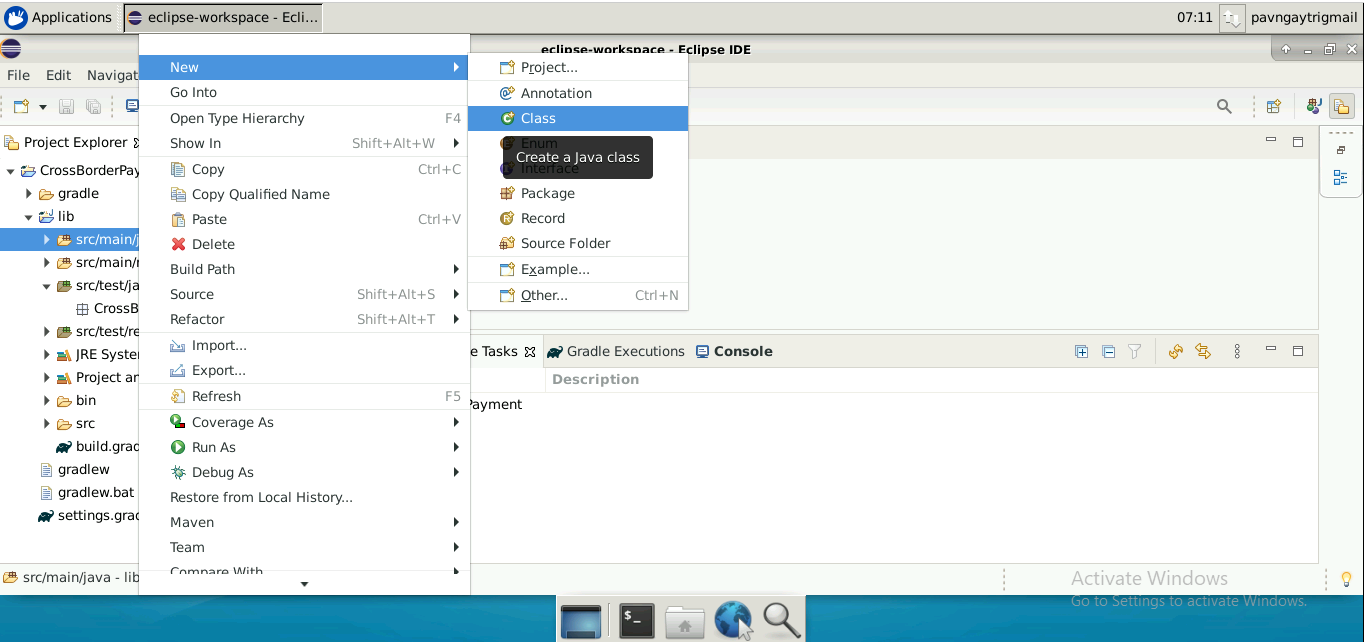
Steps to be followed:

1. Creating a LetterOfCredit.java file
2. Creating a LetterOfCreditTransfer.java file
3. Compiling the chaincode in the Terminal

**Step 1: Creating a LetterOfCreditIssue.java file**

1. Right-click on the project and navigate to **New** > **Class** to create a

**LetterOfCreditIssue.java** class



1. Add the following code in the **LetterOfCreditIssue.java** file:

***package CrossBorderPayment;***

***import org.hyperledger.fabric.contract.annotation.DataType;***

***import org.hyperledger.fabric.contract.annotation.Property;***

***import com.owlike.genson.annotation.JsonProperty;***

***import java.util.Objects;***

***@DataType()***

***public final class LetterOfCreditIssue {***

***@Property()***

***private final String id;***

***@Property()***

***private final String expiryDate;***

***@Property()***

***private final String buyer;***

***@Property()***

***private final String bank;***

***@Property()***

***private final String seller;***

***@Property()***

***private final String amount;***

***@Property()***

***private final String status;***

***public String getId() {***

***return id;***

***}***

***public String getExpiryDate() {***

***return expiryDate;***

***}***

***public String getBuyer() {***

***return buyer;***

***}***

***public String getBank() {***

***return bank;***

***}***

***public String getSeller() {***

***return seller;***

***}***

***public String getAmount() {***

***return amount;***

***}***

***public String getStatus() {***

***return status;***

***}***

***public LetterOfCreditIssue(@JsonProperty("id") final String id, @JsonProperty("expiryDate") final String expiryDate,***

***@JsonProperty("buyer") final String buyer, @JsonProperty("bank") final String bank,***

***@JsonProperty("seller") final String seller, @JsonProperty("amount") final String amount,***

***@JsonProperty("status") final String status) {***

***this.id = id;***

***this.expiryDate = expiryDate;***

***this.buyer = buyer;***

***this.bank = bank;***

***this.seller = seller;***

***this.amount = amount;***

***this.status = status;***

***}***

***@Override***

***public boolean equals(final Object obj) {***

***if (this == obj) {***

***return true;***

***}***

***if ((obj == null) || (getClass() != obj.getClass())) {***

***return false;***

***}***

***LetterOfCreditIssue other = (LetterOfCreditIssue) obj;***

***return Objects.deepEquals(new String[]{getId(), getExpiryDate(), getBuyer(), getBank(), getSeller(), getAmount(), getStatus()},***

***new String[]{other.getId(), other.getExpiryDate(), other.getBuyer(), other.getBank(), other.getSeller(), other.getAmount(), other.getStatus()});***

***}***

***@Override***

***public int hashCode() {***

***return Objects.hash(getId(), getExpiryDate(), getBuyer(), getBank(), getSeller(), getAmount(), getStatus());***

***}***

***@Override***

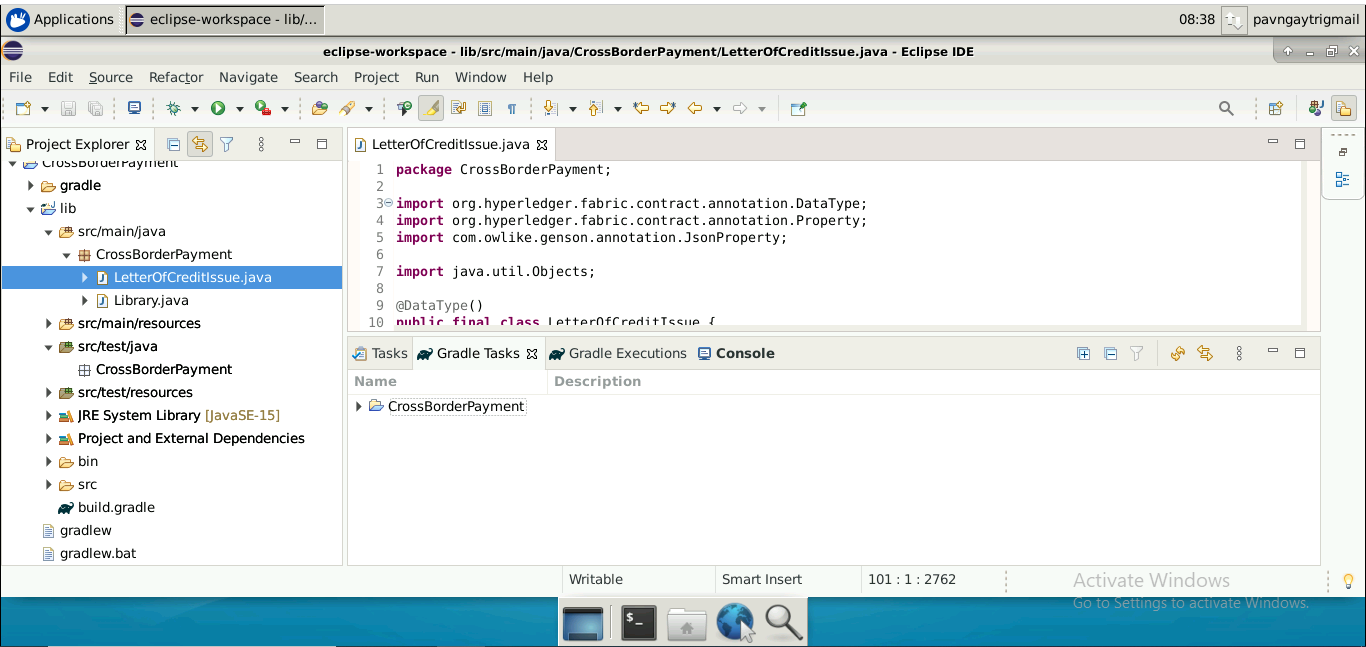
***public String toString() {***

***return this.getClass().getSimpleName() + "@" + Integer.toHexString(hashCode()) + " [id=" + id + ", expiryDate=" + expiryDate***

***+ ", buyer=" + buyer + ", bank=" + bank + ", seller=" + seller + ", amount=" + amount + ", status=" + status + "]";***

***}***

***}***



**Step 2: Creating a LetterOfCreditIssueAccept.java file**

Follow Step 1.1 to create a **LetterOfCreditIssueAccept.java** class and add the following code in it:

***package CrossBorderPayment;***

***import org.hyperledger.fabric.contract.Context;***

***import org.hyperledger.fabric.contract.ContractInterface;***

***import org.hyperledger.fabric.contract.annotation.Contract;***

***import org.hyperledger.fabric.contract.annotation.Default;***

***import org.hyperledger.fabric.contract.annotation.Info;***

***import org.hyperledger.fabric.contract.annotation.Transaction;***

***import org.hyperledger.fabric.shim.ChaincodeException;***

***import org.hyperledger.fabric.shim.ChaincodeStub;***

***import com.owlike.genson.Genson;***

***@Contract(***

***name = "LetterOfCreditIssueAccept",***

***info = @Info(***

***title = "Letter of Credit Issue Acceptance contract",***

***description = "A Sample LetterOfCreditIssue accept chaincode example",***

***version = "0.0.1-SNAPSHOT"***

***)***

***)***

***@Default***

***public final class LetterOfCreditIssueAccept implements ContractInterface {***

***private final Genson genson = new Genson();***

***private enum CrossBorderPaymentErrors {***

***LETTER\_OF\_CREDIT\_ISSUE\_NOT\_FOUND,***

***LETTER\_OF\_CREDIT\_ISSUE\_ALREADY\_EXISTS***

***}***

***@Transaction()***

***public void initLedger(final Context ctx) {***

***ChaincodeStub stub = ctx.getStub();***

***LetterOfCreditIssue letterOfCreditIssue = new LetterOfCreditIssue("1", "20thApril2024", "Jack", "INTERNATIONAL\_TRADE\_BANK", "Sundaram", "80,000$", "Issued");***

***String letterOfCreditIssueState = genson.serialize(letterOfCreditIssue);***

***stub.putStringState("1", letterOfCreditIssueState);***

***}***

***@Transaction()***

***public LetterOfCreditIssue addNewLetterOfCreditIssue(final Context ctx, final String id, final String expiryDate,***

***final String buyer, final String bank,***

***final String seller, final String amount,***

***final String status) {***

***ChaincodeStub stub = ctx.getStub();***

***String letterOfCreditIssueState = stub.getStringState(id);***

***if (!letterOfCreditIssueState.isEmpty()) {***

***String errorMessage = String.format("LetterOfCreditIssue %s already exists", id);***

***System.out.println(errorMessage);***

***throw new ChaincodeException(errorMessage, CrossBorderPaymentErrors.LETTER\_OF\_CREDIT\_ISSUE\_ALREADY\_EXISTS.toString());***

***}***

***LetterOfCreditIssue letterOfCreditIssue = new LetterOfCreditIssue(id, expiryDate, buyer, bank, seller, amount, status);***

***letterOfCreditIssueState = genson.serialize(letterOfCreditIssue);***

***stub.putStringState(id, letterOfCreditIssueState);***

***return letterOfCreditIssue;***

***}***

***@Transaction()***

***public LetterOfCreditIssue changeLetterOfCreditIssueOwnership(final Context ctx, final String id, final String newLetterOfCreditIssueSeller) {***

***ChaincodeStub stub = ctx.getStub();***

***String letterOfCreditIssueState = stub.getStringState(id);***

***if (letterOfCreditIssueState.isEmpty()) {***

***String errorMessage = String.format("LetterOfCreditIssue %s does not exist", id);***

***System.out.println(errorMessage);***

***throw new ChaincodeException(errorMessage, CrossBorderPaymentErrors.LETTER\_OF\_CREDIT\_ISSUE\_NOT\_FOUND.toString());***

***}***

***LetterOfCreditIssue letterOfCreditIssue = genson.deserialize(letterOfCreditIssueState, LetterOfCreditIssue.class);***

***LetterOfCreditIssue newLetterOfCreditIssue = new LetterOfCreditIssue(letterOfCreditIssue.getId(), letterOfCreditIssue.getExpiryDate(),***

***letterOfCreditIssue.getBuyer(), letterOfCreditIssue.getBank(), newLetterOfCreditIssueSeller,***

***letterOfCreditIssue.getAmount(), letterOfCreditIssue.getStatus());***

***String newLetterOfCreditIssueState = genson.serialize(newLetterOfCreditIssue);***

***stub.putStringState(id, newLetterOfCreditIssueState);***

***return newLetterOfCreditIssue;***

***}***

***@Transaction()***

***public LetterOfCreditIssue queryLetterOfCreditIssueById(final Context ctx, final String id) {***

***ChaincodeStub stub = ctx.getStub();***

***String letterOfCreditIssueState = stub.getStringState(id);***

***if (letterOfCreditIssueState.isEmpty()) {***

***String errorMessage = String.format("LetterOfCreditIssue %s does not exist", id);***

***System.out.println(errorMessage);***

***throw new ChaincodeException(errorMessage, CrossBorderPaymentErrors.LETTER\_OF\_CREDIT\_ISSUE\_NOT\_FOUND.toString());***

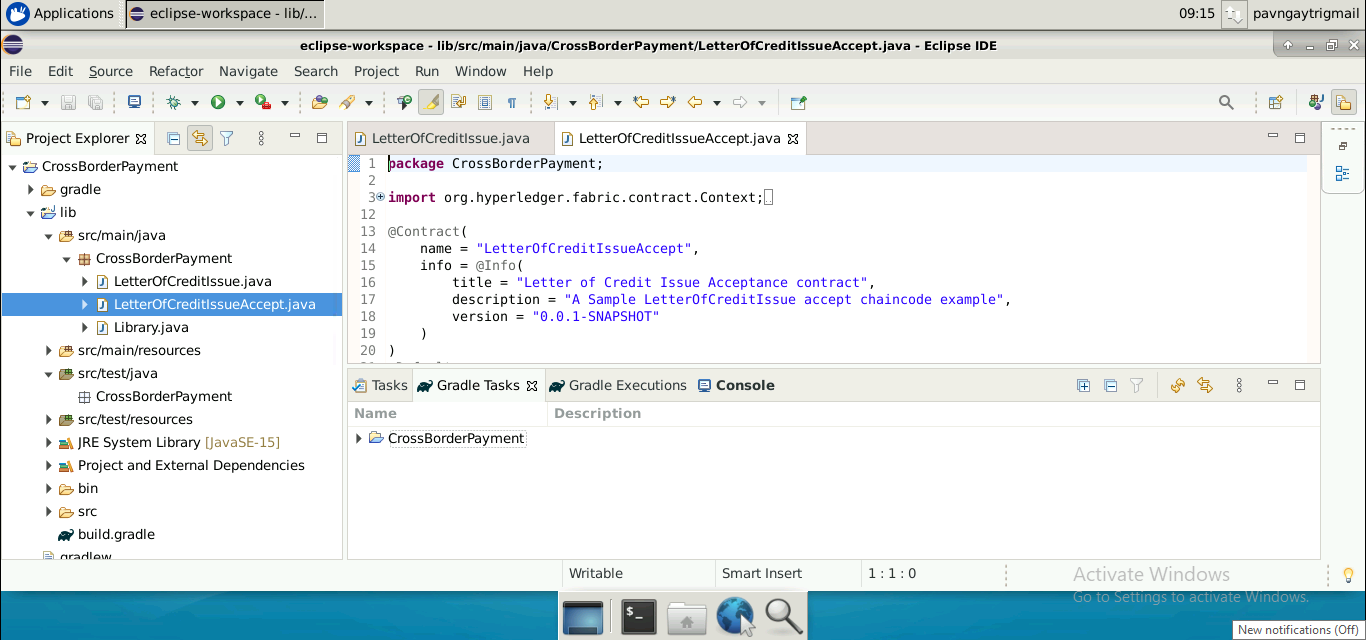
***}***

***LetterOfCreditIssue letterOfCreditIssue = genson.deserialize(letterOfCreditIssueState, LetterOfCreditIssue.class);***

***return letterOfCreditIssue;***

***}***

***}***

****

**Step 3: Compiling the project**

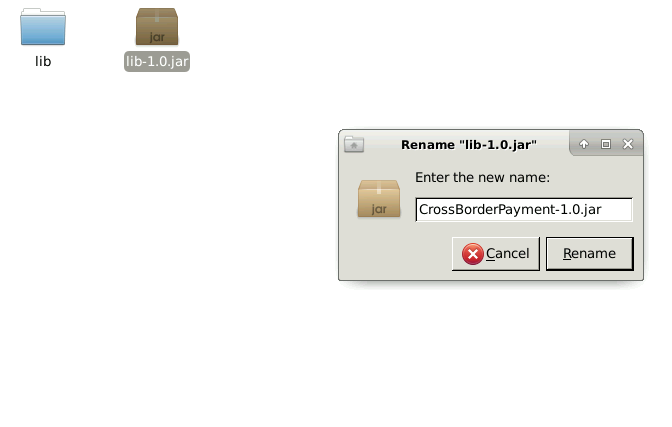
1. To compile the chaincode, run the command on the terminal:

**cd eclipse-workspace/CrossBorderPayment/**

***./gradlew installDist***

1. Rename the compiled file.Go to

eclipse-workspace/CrossBorderPayment /lib/build/install/lib/ folder



The chaincode for a Cross Border Payment is successfully created.