Creating a new gradle project for farmProduct.

**Gradle project source code**

***Project name will be farmProduct :***

***/\****

***\* This file was generated by the Gradle 'init' task.***

***\****

***\* This generated file contains a sample Java Library project to get you started.***

***\* For more details take a look at the Java Libraries chapter in the Gradle***

***\* User Manual available at https://docs.gradle.org/6.3/userguide/java\_library\_plugin.html***

***\*/***

***plugins {***

***id 'checkstyle'***

***// Apply the java-library plugin to add support for Java Library***

***id 'java-library-distribution'***

***}***

***group ‘farmProduct’***

***version '1.0'***

***repositories {***

***// Use jcenter for resolving dependencies.***

***// You can declare any Maven/Ivy/file repository here.***

***maven {***

***url "https://hyperledger.jfrog.io/hyperledger/fabric-maven"***

***}***

***jcenter()***

***maven {***

***url 'https://jitpack.io'***

***}***

***}***

***dependencies {***

***compileOnly 'org.hyperledger.fabric-chaincode-java:fabric-chaincode-shim:2.2.+'***

***implementation 'com.owlike:genson:1.5'***

***// This dependency is exported to consumers, that is to say found on their compile classpath.***

***api 'org.apache.commons:commons-math3:3.6.1'***

***// This dependency is used internally, and not exposed to consumers on their own compile classpath.***

***implementation 'com.google.guava:guava:28.2-jre'***

***// Use JUnit test framework***

***testImplementation 'junit:junit:4.12'***

***}***

***Creating a chaincode for farmProduct***

***Package name 🡪 farmproduct***

***Product.java 🡪 name of the variable file***

***package farmProduct;***

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

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

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

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

***@DataType()***

***public final class product {***

***@Property()***

***private final String productId;***

***@Property()***

***private final String productDescription;***

***@Property()***

***private final String producerName;***

***@Property()***

***private final String producerAddress;***

***@Property()***

***private final String harvestDate;***

***@Property()***

***private final String distributerName;***

***@Property()***

***private final String distributerAddress;***

***@Property()***

***private final String prodToDistDate;***

***@Property()***

***private final String retailerName;***

***@Property()***

***private final String retailerAddress;***

***@Property()***

***private final String distToRetaDate;***

***public String getProductId() {***

***return productId;***

***}***

***public String getProductDescription() {***

***return productDescription;***

***}***

***public String getProducerName() {***

***return producerName;***

***}***

***public String getProducerAddress() {***

***return producerAddress;***

***}***

***public String getHarvestDate() {***

***return harvestDate;***

***}***

***public String getDistributerName() {***

***return distributerName;***

***}***

***public String getDistributerAddress() {***

***return distributerAddress;***

***}***

***public String getProdToDistDate() {***

***return prodToDistDate;***

***}***

***public String getRetailerName() {***

***return retailerName;***

***}***

***public String getRetailerAddress() {***

***return retailerAddress;***

***}***

***public String getDistToRetaDate() {***

***return distToRetaDate;***

***}***

***public product(@JsonProperty("productId") final String productId, @JsonProperty("productDescription") final String productDescription, @JsonProperty("producerName") final String producerName,***

***@JsonProperty("producerAddress") final String producerAddress,***

***@JsonProperty("harvestDate") final String harvestDate,***

***@JsonProperty("distributerName") final String distributerName,***

***@JsonProperty("distributerAddress") final String distributerAddress,***

***@JsonProperty("prodToDistDate") final String prodToDistDate,***

***@JsonProperty("retailerName") final String retailerName,***

***@JsonProperty("retailerAddress") final String retailerAddress,***

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

***this.productId = productId;***

***this.productDescription = productDescription;***

***this.producerName = producerName;***

***this.producerAddress = producerAddress;***

***this.harvestDate = harvestDate;***

***this.distributerName = distributerName;***

***this.distributerAddress = distributerAddress;***

***this.prodToDistDate = prodToDistDate;***

***this.retailerName = retailerName;***

***this.retailerAddress = retailerAddress;***

***this.distToRetaDate = distToRetaDate;***

***}***

***@Override***

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

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

***return true;***

***}***

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

***return false;***

***}***

***product other = (product) obj;***

***return Objects.deepEquals(new String[]{getProductId(), getProductDescription(), getProducerName(), getProducerAddress(), getHarvestDate(),***

***getDistributerName(), getDistributerAddress(), getProdToDistDate(), getRetailerName(), getRetailerAddress(), getDistToRetaDate()},***

***new String[]{other.getProductId(), other.getProductDescription(), other.getProducerName(), other.getProducerAddress(), other.getHarvestDate(),***

***other.getDistributerName(), other.getDistributerAddress(), other.getProdToDistDate(), other.getRetailerName(), other.getRetailerAddress(), other.getDistToRetaDate()});***

***}***

***@Override***

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

***return Objects.hash(getProductId(), getProductDescription(), getProducerName(), getProducerAddress(), getHarvestDate(),***

***getDistributerName(), getDistributerAddress(), getProdToDistDate(), getRetailerName(), getRetailerAddress(), getDistToRetaDate());***

***}***

***@Override***

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

***return this.getClass().getSimpleName() + "@" + Integer.toHexString(hashCode()) + " [productId=" + productId + ", productDescription=" + productDescription + ", producerName=" + producerName + ", producerAddress=" + producerAddress + ", harvestDate=" + harvestDate + ", distributerName=" + distributerName + ", distributerAddress=" + distributerAddress + " , prodToDistDate=" + prodToDistDate + ", retailerName=" + retailerName + ", retailerAddress=" + retailerAddress + ", distToRetaDate=" + distToRetaDate + "]";***

***}***

***}***

Name of Function File

**productTransfer.java**

***package farmProduct;***

***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 = "farmProduct",***

***info = @Info(***

***title = "farmProduct contract",***

***description = "A Sample Farm Product transfer chaincode example",***

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

***@Default***

***public final class productTransfer implements ContractInterface {***

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

***private enum FarmProductErrors {***

***PRODUCT\_NOT\_FOUND,***

***PRODUCT\_ALREADY\_EXIST***

***}***

***@Transaction()***

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

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

***product product = new product("01", "GreenApple", "Capsicum", "USA", "12/2/2023", "S N Enterprises", "BENGALURU", "13/2/2023", "sai vegitables", "BENGALURU", "14/2/2023");***

***String productState = genson.serialize(product);***

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

***}***

***@Transaction()***

***public product addNewProduct(final Context ctx, final String productId, final String productDescription, final String producerName, final String producerAddress, final String harvestDate) {***

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

***String productState = stub.getStringState(productId);***

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

***String errorMessage = String.format("Product already exists with ID: %s", productId);***

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

***throw new ChaincodeException(errorMessage, FarmProductErrors.PRODUCT\_ALREADY\_EXIST.toString());***

***}***

***product product = new product(productId, productDescription, producerName, producerAddress, harvestDate, productState, productState, productState, productState, productState, productState);***

***productState = genson.serialize(product);***

***stub.putStringState(productId, productState);***

***return product;***

***}***

***@Transaction()***

***public product queryProductById(final Context ctx, final String productId) {***

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

***String productState = stub.getStringState(productId);***

***if (productState.isEmpty()) {***

***String errorMessage = String.format("Product with ID %s does not exist", productId);***

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

***throw new ChaincodeException(errorMessage, FarmProductErrors.PRODUCT\_NOT\_FOUND.toString());***

***}***

***return genson.deserialize(productState, product.class);***

***}***

***@Transaction()***

***public String tranferOwnershipProducerToDistributor(final Context ctx, final String productId, final String distributorName, final String distributorAddress, final String prodToDistDate) {***

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

***String productState = stub.getStringState(productId);***

***if (productState.isEmpty()) {***

***String errorMessage = String.format("Product with ID %s does not exist", productId);***

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

***throw new ChaincodeException(errorMessage, FarmProductErrors.PRODUCT\_NOT\_FOUND.toString());***

***}***

***product product = genson.deserialize(productState, product.class);***

***product newProduct = new product(product.getProductId(), product.getProductDescription(), distributorName, distributorAddress, prodToDistDate, productState, prodToDistDate, distributorAddress, distributorName, null, productState);***

***String newProductState = genson.serialize(newProduct);***

***stub.putStringState(productId, newProductState);***

***return productId;***

***}***

***@Transaction()***

***public String tranferOwnershipDistributorToRetailer(final Context ctx, final String productId, final String retailerName, final String retailerAddress, final String distToRetaDeta) {***

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

***String productState = stub.getStringState(productId);***

***if (productState.isEmpty()) {***

***String errorMessage = String.format("Product with ID %s does not exist", productId);***

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

***throw new ChaincodeException(errorMessage, FarmProductErrors.PRODUCT\_NOT\_FOUND.toString());***

***}***

***product product = genson.deserialize(productState, product.class);***

***product newProduct = new product(product.getProductId(), product.getProductDescription(), retailerName, retailerAddress, distToRetaDeta, productState, distToRetaDeta, retailerAddress, retailerName, productId, productState);***

***String newProductState = genson.serialize(newProduct);***

***stub.putStringState(productId, newProductState);***

***return productId;***

***}***

***}***

**Commands used invoke the functions:**

**Add new product**

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls $CORE\_PEER\_TLS\_ENABLED --cafile $ORDERER\_CA -C mychannel -n farmProduct --peerAddresses localhost:7051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG1 --peerAddresses localhost:9051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG2 -c '{"Args":["addNewProduct","03","Green chilli","BABA Vegitables","USA","13/3/2024"]}'

**product transfer from producer to distributer**

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls $CORE\_PEER\_TLS\_ENABLED --cafile $ORDERER\_CA -C mychannel -n farmProduct -- peerAddresses localhost:7051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG1 --peerAddresses localhost:9051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG2 -c '{"Args":["tranferOwnershipProducerToDistributor", "03", "<GURU ENTERPRISES>", "<BENGALURU>", "<16/3/2024>"]}'

**product transfer from distributer to producer**

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls $CORE\_PEER\_TLS\_ENABLED --cafile $ORDERER\_CA -C mychannel -n farmProduct --peerAddresses localhost:7051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG1 --peerAddresses localhost:9051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG2 -c '{"Args":["tranferOwnershipDistributorToRetailer", "03", "Green THOTA", "BENGALURU", "20/3/2024"]}'

**full supply chain information of product**

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls $CORE\_PEER\_TLS\_ENABLED --cafile $ORDERER\_CA -C mychannel -n farmProduct --peerAddresses localhost:7051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG1 --peerAddresses localhost:9051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE\_ORG2 -c '{"Args":["queryProductById", "03"]}'