cancelFood (UID, serveDate, meal){

u = User.find(UID);

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true){

reservedFood = Reserve.find(UID, serveDate, meal);

if (reservedFood == null){

return("unsuccessful");

} else {

result = Reserve.delete(reservedFood);

if(result == true)

return("successful");

else

return("unsuccessful");

}

} else {

return("unsuccessful");

}

}

reserveFood (UID, serveDate, meal){

u = User.find(UID);

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true){

result = Reserve.insert (UID, serveDate, meal);

if (result){

return("successful");

} else {

return("unsuccessful ");

}

} else {

return("unsuccessful ");

}

}

increaseCredit (UID, credit) {

u = User.find(UID);

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true){

bankT = makeBankTransaction(credit);

if(bankT){

u.credit += credit ;

result = IncreaseCreditLog.create(UID, credit) & User.update(UID);

if(result)

return ("successful");

else

return ("unsuccessful");

} else {

return ("unsuccessful");

}

} else {

return ("unsuccessful");

}

}

transferCredit (srcUID , destUID , amount) {

src = User.find(srcUID);

dst = User.find(destUID);

src.lastActivity = getCurrentDateTime();

bool = User.update(src);

if(bool == true) {

valid = src == null || dst == null || src.credit < amount;

if (valid == true){

return ("unsuccessful");

} else {

src.credit -= amount;

dst.credit += amount;

result = User.update(src)

&User.update(dst)

&TransferCreditLog.insert(src, dst, amount);

if(result == true)

return ("successful");

else

return ("unsuccessful");

}

} else {

return ("unsuccessful");

}

}

getActivityReport(UID , startDate, endDate , reportType){

u = User.find(UID);

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true){

if (reportType == increase){

logTable = IncreaseCreditLog.findMatching(UID, startDate, endDate);

} else {

logTable = TransferCreditLog.findMatching(UID, startDate, endDate);

}

return logTable;

} else {

return (“unsuccessful”);

}

}

getFoodSchedule(UID, weekStartDate, weekEndDate){

u = User.find(UID);

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool = true) {

g = u.gender;

foodPlanTable = Serve.findMatching(UID, weekStartDate, weekEndDate, g);

return foodPlanTable;

} else {

return(“unsuccessful”);

}

}

login( UID , password ){

u = User.find ( UID );

if (u == null) {

return(“unsuccessful”);

} else {

if (u.password == password) {

u.isLogin = true ;

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if (bool == true){

type = getUserType(UID);

return type;

}else

return ("unsuccessful");

} else {

return(“unsuccessful”);

}

}

}

logout(UID){

u = User.find(UID);

if (u == null || !u.isLogin){

return("unsuccessful!");

} else {

u.isLogin = false;

bool = User.update(u);

if (bool == true)

return("successful!");

else

return("unsuccessful!");

}

}

registerUser(UID, password, name, gender){

u = User.findAdmin();

if (u != null){

u.lastActivity = getCurrentDateTime();

result = User.update(u) & User.insert(UID , password , name , gender);

if(result == true)

return(“successful”);

else

return(“unsuccessful”);

} else {

return (“unsuccessful”);

}

}

determineFood(foodName, price){

u = User.findTheKitchen();

if(u != null) {

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if (bool == true) {

f = Food.find(foodName);

if (f == null){

result = Food.insert(foodName, price, false);

if(result == true)

return ("new food inserted successfully");

else

return ("unsuccessful");

} else {

result = Food.update(foodName, price, false);

if(result == true)

return ("food updated successfully");

else

return ("unsuccessful");

}

} else {

return(“unsuccessful”);

}

} else {

return(“unsuccessful”);

}

}

determineFoodPlan(foodName , serveDate , meal){

u = User.findTheKitchen();

if (u != null) {

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true) {

f = Food.find(foodName);

if ( f == null ) {

return("unsuccessful!");

} else {

result = Serve.insert(foodName, serveDate, meal);

if (result == true)

return("successful");

else

return("unsuccessful!");

}

} else {

return (“unsuccessful);

}

} else {

return (“unsuccessful”);

}

}

foodStatisticsSearch(serveDate, meal , gender ){

u = User.findTheKitchen();

if (u == null) {

return (“unsuccessful”);

} else {

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if ( bool == true) {

reserveTable = Reserve.findMatching( serve\_date, meal , gender );

return reserveTable;

} else {

return (“unsuccessful”);

}

}

}

getFoodStatistics(startServeDate, endServeDate){

u = User.findTheKitchen();

if( u == null) {

return (“unsuccessful”);

} else {

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true) {

reserveTable = Reserve.findMatching(startServeDate, endServeDate);

return reserveTable;

} else {

return (“unsuccessful”);

}

}

}

kickOut () {

users[] = User.findAll().where(isLogin == true);

counter = 0;

foreach(u : users) {

if (u.lastActivity – currentDateTime() > 12)

u.isLogin = false;

result = User.update(u);

if (result == true)

counter++;

}

return counter;

}

getNewFoodPrice( request ) {

if(request){

u = User.findAdmin();

if (u != null){

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true)

foodList = Food.findAll ( ).where (isApproved == false);

return foodList;

else

return(“unsuccessful”);

} else {

return (“unsuccessful”);

}

}else{

Return (‘unsuccessful”)

}

}

approveNewFoodPrice (priceApprovementList) {

u = User.findAdmin();

if (u != null){

u.lastActivity = getCurrentDateTime();

bool = User.update(u);

if(bool == true)

index = 0;

foreach (approve : priceApprovementList) {

if (approve == true) {

foodList[index].isApproved = true;

}

index++;

}

result = Food.update(foodList);

if (result == true) {

return (“successful”);

} else {

return (“unsuccessful”);

}

else

return(“unsuccessful”);

} else {

return (“unsuccessful”);

}

}