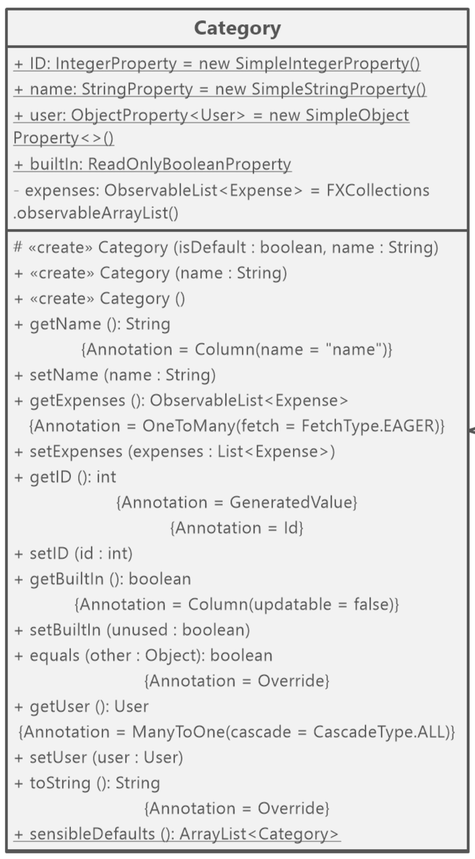
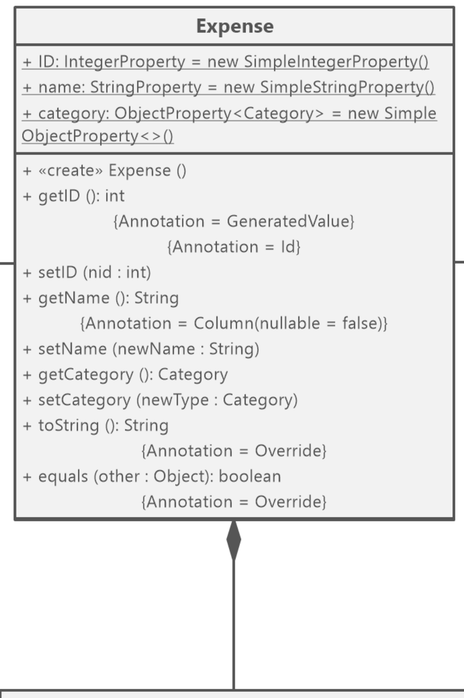
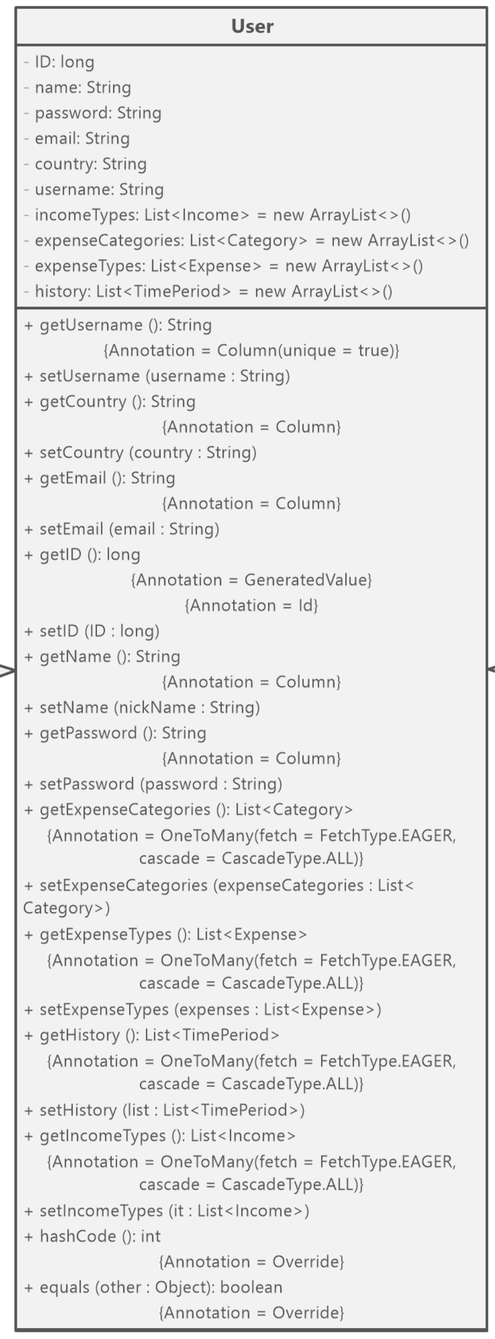
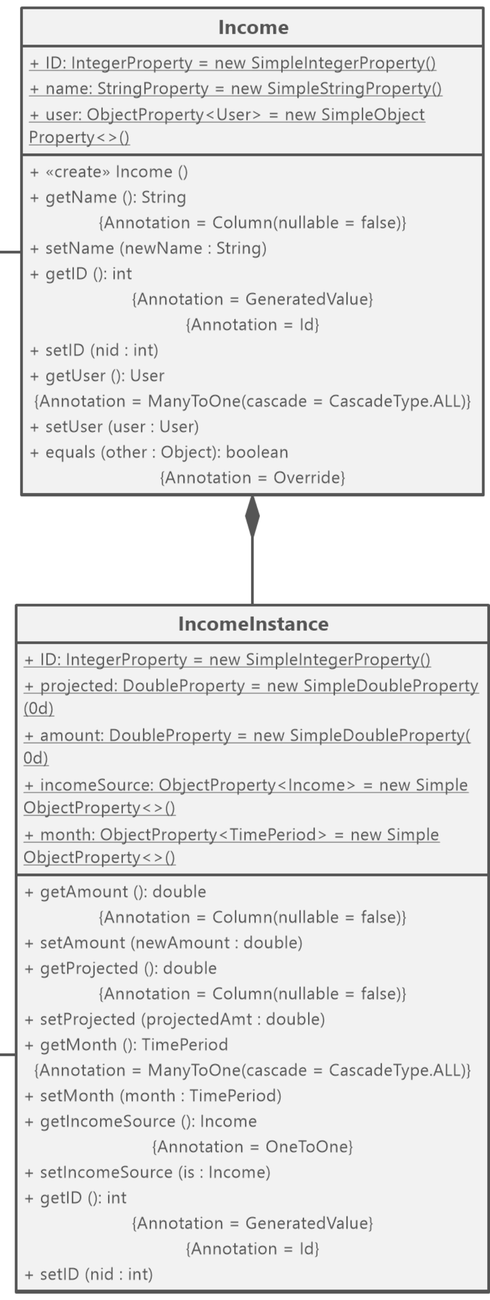
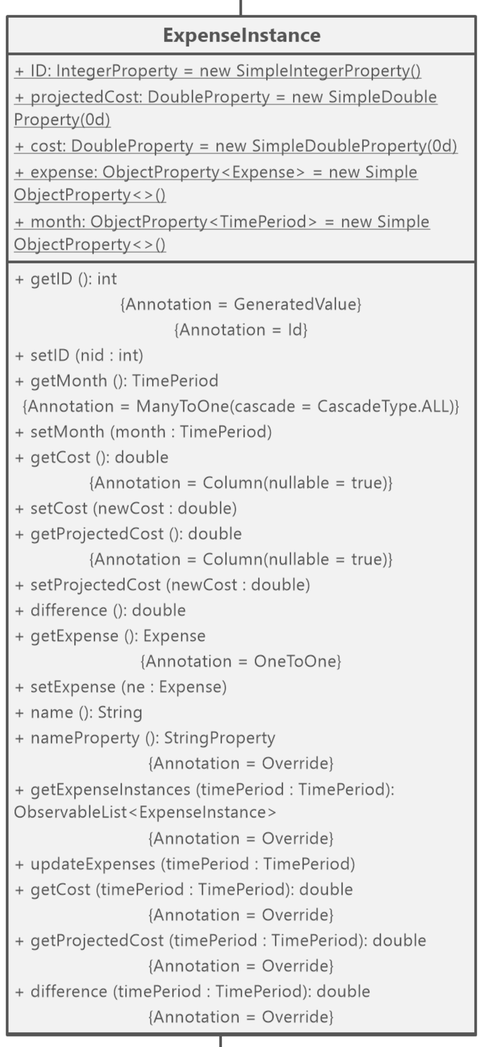

Category
+ ID: IntegerProperty = new SimpleIntegerProperty()
+ name: StringProperty = new SimpleStringProperty()
+ user: ObjectProperty<User> = new SimpleObject Property<>()
+ builtIn: ReadOnlyBooleanProperty
- expenses: ObservableList<Expense> = FXCollections .observableArrayList()
«create» Category (isDefault : boolean, name : String)
+ «create» Category (name : String)
+ «create» Category ()
+ getName (): String
{Annotation = Column(name = "name")}
+ setName (name : String)
+ getExpenses (): ObservableList<Expense>
{Annotation = OneToMany(fetch = FetchType.EAGER)}
+ setExpenses (expenses : List<Expense>)
+ getID (): int
{Annotation = GeneratedValue}
{Annotation = Id}
+ setID (id : int)
+ getBuiltIn (): boolean
{Annotation = Column(updatable = false)}
+ setBuiltIn (unused : boolean)
+ equals (other : Object): boolean
{Annotation = Override}
+ getUser (): User
{Annotation = ManyToOne(cascade = CascadeType.ALL)}
+ setUser (user : User)
+ toString (): String
{Annotation = Override}
+ sensibleDefaults (): ArrayList<Category>

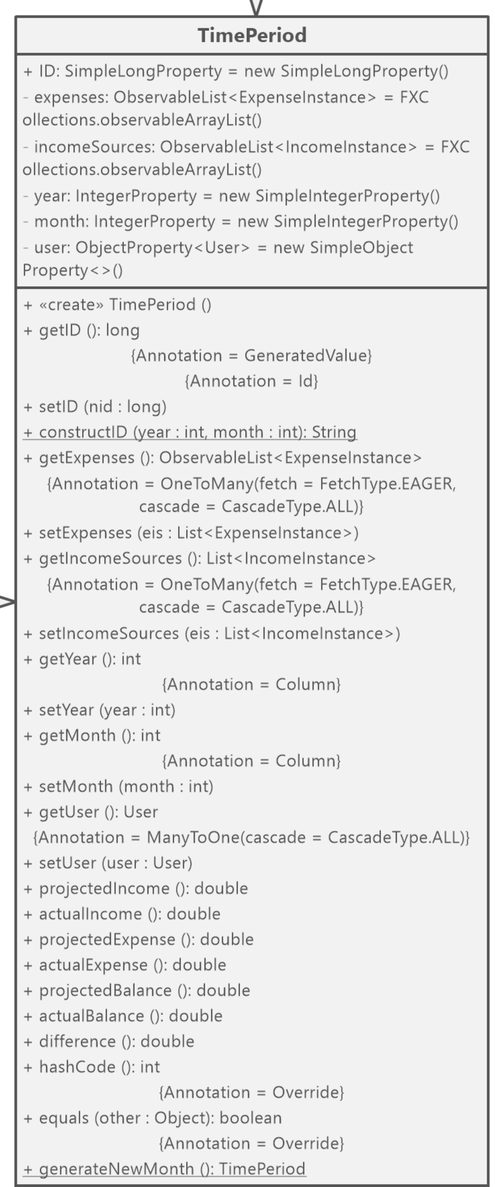

Expense
+ ID: IntegerProperty = new SimpleIntegerProperty()
+ name: StringProperty = new SimpleStringProperty()
+ category: ObjectProperty<Category> = new Simple ObjectProperty<>()
+ «create» Expense ()
+ getID (): int
{Annotation = GeneratedValue}
{Annotation = Id}
+ setID (nid : int)
+ getName (): String
{Annotation = Column(nullable = false)}
+ setName (newName : String)
+ getCategory (): Category
+ setCategory (newType : Category)
+ toString (): String
{Annotation = Override}
+ equals (other : Object): boolean
{Annotation = Override}


User
- ID: long
- name: String
- password: String
- email: String
- country: String
- username: String
- incomeTypes: List<Income> = new ArrayList<>()
- expenseCategories: List<Category> = new ArrayList<>()
- expenseTypes: List<Expense> = new ArrayList<>()
- history: List<TimePeriod> = new ArrayList<>()
+ getUsername (): String
{Annotation = Column(unique = true)}
+ setUsername (username : String)
+ getCountry (): String
{Annotation = Column}
+ setCountry (country : String)
+ getEmail (): String
{Annotation = Column}
+ setEmail (email : String)
+ getID (): long
{Annotation = GeneratedValue}
{Annotation = Id}
+ setID (ID : long)
+ getName (): String
{Annotation = Column}
+ setName (nickName : String)
+ getPassword (): String
{Annotation = Column}
+ setPassword (password : String)
+ getExpenseCategories (): List<Category>
{Annotation = OneToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)}
+ setExpenseCategories (expenseCategories : List< Category>)
+ getExpenseTypes (): List<Expense>
{Annotation = OneToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)}
+ setExpenseTypes (expenses : List<Expense>)
+ getHistory (): List<TimePeriod>
{Annotation = OneToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)}
+ setHistory (list : List<TimePeriod>)
+ getIncomeTypes (): List<Income>
{Annotation = OneToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)}
+ setIncomeTypes (it : List<Income>)
+ hashCode (): int
{Annotation = Override}
+ equals (other : Object): boolean
{Annotation = Override}


Income
+ ID: IntegerProperty = new SimpleIntegerProperty()
+ name: StringProperty = new SimpleStringProperty()
+ user: ObjectProperty<User> = new SimpleObject Property<>()
+ «create» Income ()
+ getName (): String
{Annotation = Column(nullable = false)}
+ setName (newName : String)
+ getID (): int
{Annotation = GeneratedValue}
{Annotation = Id}
+ setID (nid : int)
+ getUser (): User
{Annotation = ManyToOne(cascade = CascadeType.ALL)}
+ setUser (user : User)
+ equals (other : Object): boolean
{Annotation = Override}


ExpenseInstance
+ ID: IntegerProperty = new SimpleIntegerProperty()
+ projectedCost: DoubleProperty = new SimpleDouble Property(0d)
+ cost: DoubleProperty = new SimpleDoubleProperty(0d)
+ expense: ObjectProperty<Expense> = new Simple ObjectProperty<>()
+ month: ObjectProperty<TimePeriod> = new Simple ObjectProperty<>()
+ getID (): int
{Annotation = GeneratedValue}
{Annotation = Id}
+ setID (nid : int)
+ getMonth (): TimePeriod
{Annotation = ManyToOne(cascade = CascadeType.ALL)}
+ setMonth (month : TimePeriod)
+ getCost (): double
{Annotation = Column(nullable = true)}
+ setCost (newCost : double)
+ getProjectedCost (): double
{Annotation = Column(nullable = true)}
+ setProjectedCost (newCost : double)
+ difference (): double
+ getExpense (): Expense
{Annotation = OneToOne}
+ setExpense (ne : Expense)
+ name (): String
+ nameProperty (): StringProperty
{Annotation = Override}
+ getExpenseInstances (timePeriod : TimePeriod): ObservableList<ExpenseInstance>
{Annotation = Override}
+ updateExpenses (timePeriod : TimePeriod)
+ getCost (timePeriod : TimePeriod): double
{Annotation = Override}
+ getProjectedCost (timePeriod : TimePeriod): double
{Annotation = Override}
+ difference (timePeriod : TimePeriod): double
{Annotation = Override}

IncomeInstance
+ ID: IntegerProperty = new SimpleIntegerProperty()
+ projected: DoubleProperty = new SimpleDoubleProperty (0d)
+ amount: DoubleProperty = new SimpleDoubleProperty(0d)
+ incomeSource: ObjectProperty<Income> = new Simple ObjectProperty<>()
+ month: ObjectProperty<TimePeriod> = new Simple ObjectProperty<>()
+ getAmount (): double
{Annotation = Column(nullable = false)}
+ setAmount (newAmount : double)
+ getProjected (): double
{Annotation = Column(nullable = false)}
+ setProjected (projectedAmt : double)
+ getMonth (): TimePeriod
{Annotation = ManyToOne(cascade = CascadeType.ALL)}
+ setMonth (month : TimePeriod)
+ getIncomeSource (): Income
{Annotation = OneToOne}
+ setIncomeSource (is : Income)
+ getID (): int
{Annotation = GeneratedValue}
{Annotation = Id}
+ setID (nid : int)


TimePeriod
+ ID: SimpleLongProperty = new SimpleLongProperty()
- expenses: ObservableList<ExpenseInstance> = FXC ollections.observableArrayList()
- incomeSources: ObservableList<IncomeInstance> = FXC ollections.observableArrayList()
- year: IntegerProperty = new SimpleIntegerProperty()
- month: IntegerProperty = new SimpleIntegerProperty()
- user: ObjectProperty<User> = new SimpleObject Property<>()
+ «create» TimePeriod ()
+ getID (): long
{Annotation = GeneratedValue}
{Annotation = Id}
+ setID (nid : long)
+ constructID (year : int, month : int): String
+ getExpenses (): ObservableList<ExpenseInstance>
{Annotation = OneToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)}
+ setExpenses (eis : List<ExpenseInstance>)
+ getIncomeSources (): List<IncomeInstance>
{Annotation = OneToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)}
+ setIncomeSources (eis : List<IncomeInstance>)
+ getYear (): int
{Annotation = Column}
+ setYear (year : int)
+ getMonth (): int
{Annotation = Column}
+ setMonth (month : int)
+ getUser (): User
{Annotation = ManyToOne(cascade = CascadeType.ALL)}
+ setUser (user : User)
+ projectedIncome (): double
+ actualIncome (): double
+ projectedExpense (): double
+ actualExpense (): double
+ projectedBalance (): double
+ actualBalance (): double
+ difference (): double
+ hashCode (): int
{Annotation = Override}
+ equals (other : Object): boolean
{Annotation = Override}
+ generateNewMonth (): TimePeriod