

**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА**

**ФАКУЛТЕТ ПО ИЗЧИСЛИТЕЛНА ТЕХНИКА И АВТОМАТИЗАЦИЯ**

**КАТЕДРА „СОФТУЕРНИ И ИНТЕРНЕТ ТЕХНОЛОГИИ“**

**Специалност „Софтуерни и интернет технологии“**

**Проект по Обектно – ориентирано програмиране част 2**

**Тема 8**

**Производство на вина**

**Изготвили:**

Цветелин Петров и Симеон Стойнев

(СИТ, гр. 1б, Фак. №: 19621795, 19621789)

**ВАРНА**

**2022г.**

**Условие на проекта**

**VIII. Производство на вина**   
  
Да се разработи информационна система, предоставяща възможност за организиране на производството на различни типове вино.

Системата поддържа три вида потребители администратор,оператори   
(винопроизводители) и домакин на склада с различни роли за достъп до   
функционалностите в системата.   
  
Операции за работа с потребители:

• създаване на оператори от администратор;   
• създаване на домакини от администратор;

Системата поддържа операции за винопроизводство:

• регистриране на сорт грозде и количеството му в склада(сортовете се делят на две категории - бели и черни);   
• дефиниране на количеството вино което може да се получи от килограм грозде от даден сорт;   
• регистриране на видове и количество бутилки за бутилиране на вино (750, 375, 200 и 187ml);   
• всеки тип вино се състои от един или няколко сорта и за направата му е необходимо определено количество от тях;   
• дефиниране на необходимите количества грозде за производството на даден вид вино;   
• бутилиране на виното в бутилки (бонус – системата автоматично да изчислява оптимално запълване на бутилки);

Системата поддържа Справки по произволен период за:

• наличности на сортове грозде;   
• наличност на видове бутилки;   
• наличности на видовете бутилирани вина;

Системата поддържа Известия за:

• критичен минимум и липса на даден сорт грозде;   
• критичен минимум и липса на даден вид бутилки;

**Анализ на проблема**

Системата трябва да поддържа потребители с 3 различни нива на достъп както следва: администратори, оператори и домакини на склада. Всяка роля трябва да позволява или да ограничава различните действия в програмата.

Администраторската роля на достъп трябва да позволява пълен контрол над системата и възможностите и, а именно:

* създаване на нови потребители от всеки един тип;
* регистриране на вид бутилка и нейното количество;
* регистриране на цветове грозде;
* регистриране на сортове грозде, тяхното количество в склада и количеството вино което се произвежда от един килограм от тях;
* регистриране на ново вино от един или няколко сорта грозде;
* регистриране на цикъл на производство за съответно избраното вино;
* проверка на налични количества бутилки;
* проверка на налични количества грозде;
* проверка на налични количества вино;
* проверка на вече завършени производсвени цикли;
* изтриване на потребители (от всяко ниво);
* изтриване на бутилки;
* изтриване на цветове грозде;
* изтриване на сортове грозде;
* изтриване на вина;
* изтриване на производствени цикли;
* обновяване на количество бутилки;
* обновяване на количество грозде;
* обновяване на количество вино;
* проверка за критична наличност на бутилки;
* проверка за критична наличност на сорт грозде;

Потребител с роля „оператор” би следвало да има достъп до следната функционалност в системата:

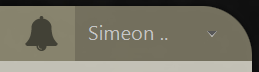
* регистриране на ново вино от един или няколко сорта грозде;
* регистриране на цикъл на производство за съответно избраното вино;
* проверка на налични количества бутилки;
* проверка на налични количества грозде;
* проверка на налични количества вино;
* проверка на вече завършени производствени цикли;
* изтриване на вина;
* изтриване на производствени цикли;
* обновяване на количество вино;
* проверка за критична наличност на бутилки;
* проверка за критична наличност на сорт грозде;

И третата роля в системата - „домакин на склад” ще има достъп до следните възможности на програмата:

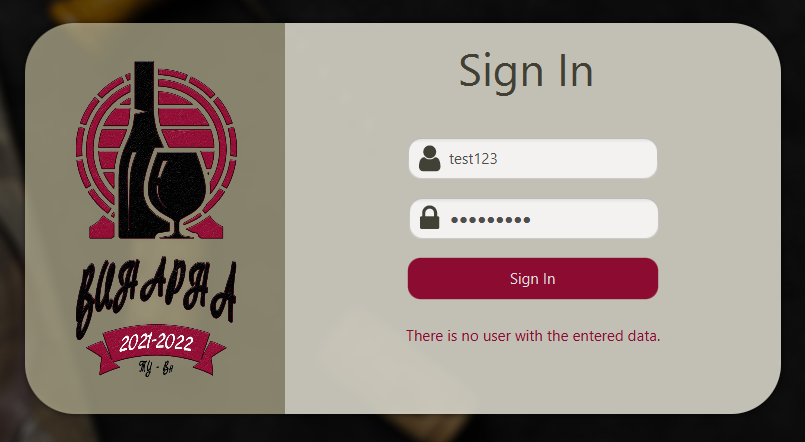
* регистриране на вид бутилка и нейното количество;
* регистриране на цветове грозде;
* регистриране на сортове грозде, тяхното количество в склада и количеството вино което се произвежда от един килограм от тях;
* проверка на налични количества бутилки;
* проверка на налични количества грозде;
* проверка на налични количества вино;
* изтриване на бутилки;
* изтриване на типове грозде;
* изтриване на сортове грозде;
* изтриване на вина;
* обновяване на количество бутилки;
* обновяване на количество грозде;
* проверка за критична наличност на бутилки;
* проверка за критична наличност на сорт грозде;

**Работа с потребителите:**

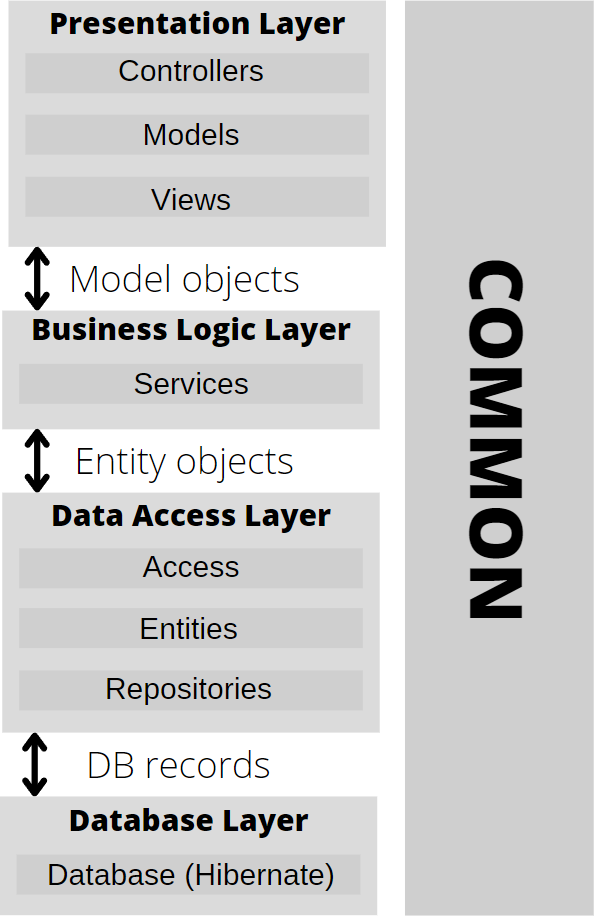
При успешно влизане в системата, програмата следи текущия вписан потребител, като извежда името му в горен десен ъгъл на потребителския интерфейс. Системата също така поддържа и известия, които се появяват в профила на всеки потребител. Известията (модул notifications) се изпращат при достигане на критичен минимум, липса на бутилка от даден вид, както и на даден сорт грозде. Те биха се появили на потребителя до името му в горен десен ъгъл, където е иконата за известия (камбанка).



При вписване на потребител се извършва проверка за коректност на данните и при липса на съвпадение се извежда известие, което уведомява потребителя за отказания достъп. Методът в системата проверява за наличност на въведеното потребителско име и парола (хеширана с md5). При успешен логин, на потребителят му бива предоставен достъп до функционалностите на системата чрез интуитивен интерфейс.



**Структура на проекта**



За ефективното изпълнение на заданието е използвана многослойна архитектура, разделена от различните слоеве, представени на горната снимка. Целта на многослойната архитектура е ясното разграничение на модулите в системата, както и по-лесна имплементация на нови функционалности и подобрения по системата.

**Presentation Layer:** изграден е от три модула – Controllers, Models и Views. В модул views се намират javafx файлове, които изграждат потребителския интерфейс и дизайна на системата. Те съдържат основните компоненти на прозорците като диалогови кутии, бутони и т.н. В модул controllers се намират класовете, които осъществяват връзката между въведените от потребителя данни и тяхната обработка в бизнес логиката. Също така те управляват отделните елементи в javafx файловете. В модул models се намират обектите в които се записва въведената от потребителя информация, която се предава на бизнес логиката. Чрез тях се постига индиректната връзка между данните от потребителя и записите в базата данни.

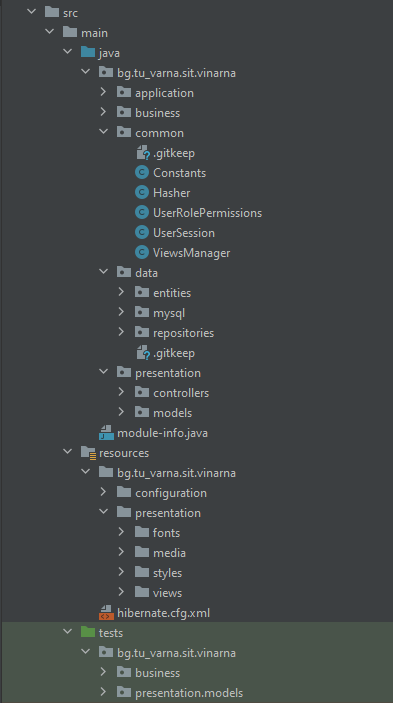
**Business Logic Layer:** обработва информацията и реализира отделните функционални изисквания на системата. Той преобразува моделите в entity обекти, които се използват от слоя за връзка с базата данни.

**Data Access Layer:** разделя се на три модула – Access, Entities и Repositories. В модул access се осъществява връзката с базата данни, чрез която се достъпват отделните записи. Модул entities съдържа класове съответстващи на таблиците от базата данни и връзките между тях. Във всеки един клас са реализирани методи, които позволяват достъпването, сравнението и визуализацията на отделните записи от съответната таблица. В модул repositories се намират класове в които се имплементират различните заявки към базата данни.

**Database Layer:** Връзката с базата данни е реализирана чрез Hibernate. Избран е този подход поради автоматичното генериране на заявки от Hibernate, както и сравнително лесната му имплементация. За реализация на базата данни е използван MySQL.

**Common Layer:** са поместени файловете, които се използват за всеобщото използване на важни класове и функционалности в системата. Такива класове са: Constants, Hasher (за md5 пароли), UserRolePermissions (създаване на лесен конфиг за достъпите на ролите в системата), UserSession и ViewsManager.

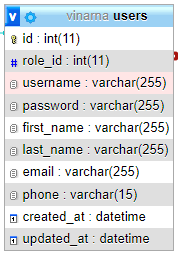
На прикачената снимка по-долу се вижда ясно цялата структура и файлово дърво на проекта:



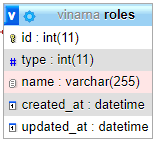
**Проектиране на системата**

**База от данни:**

В таблица users се записват всички потребители в системата. Полетата са както е представено на снимката:



В таблица roles се поместват различните роли на потребителите, като връзката е едно към едно user.role\_id = role.id.



В таблица notifications се записват всички известия, които потребителя получава, а именно достигане на критичен минимум и липса на даден сорт грозде, както и критичен минимум и липса на даден вид бутилки.

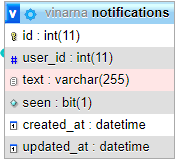
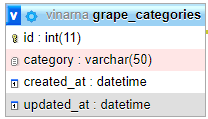
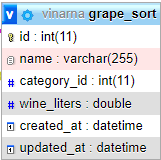


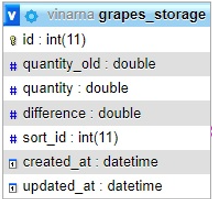
Таблица grape\_categories, в която се записват всички категории възможни за грозде - в нашия случай това са бяло и черно (по задание).



В таблица grape\_sort се записват всички сортове грозде по име, като към всеки сорт трябва да бъде зададена категория (връзка много към едно с grape\_categories).



В таблица grapes\_storage се съхраняват всички записи за складираното грозде, като към всеки запис, трябва да бъде зададен сорт грозде (връзка много към едно с grape\_sort).



Таблици wine\_types и wine\_recipes, които отговарят за типовете вино и необходимата връзка между сорт грозде - вино. Чрез тези таблици става ясно за кое вино точно колко и какъв сорт грозде е нужен. Всяка рецепта за вино има връзка много към едно с wine\_types и grape\_sort.

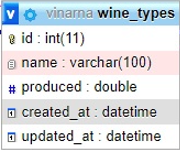
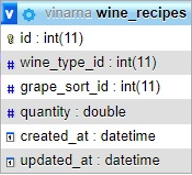
 

Таблица bottle\_types, в която се съхраняват всички видове бутилки. В случая те по задание са: 750, 375, 200 и 187мл.

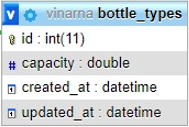
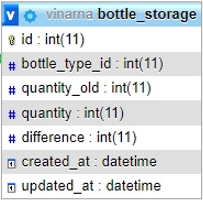
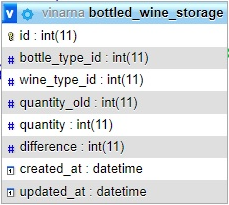


Таблица bottle\_storage, където се записват складираните бутилки. Фигурира връзка много към едно с bottle\_types, за да се знае какъв тип бутилки се складират.

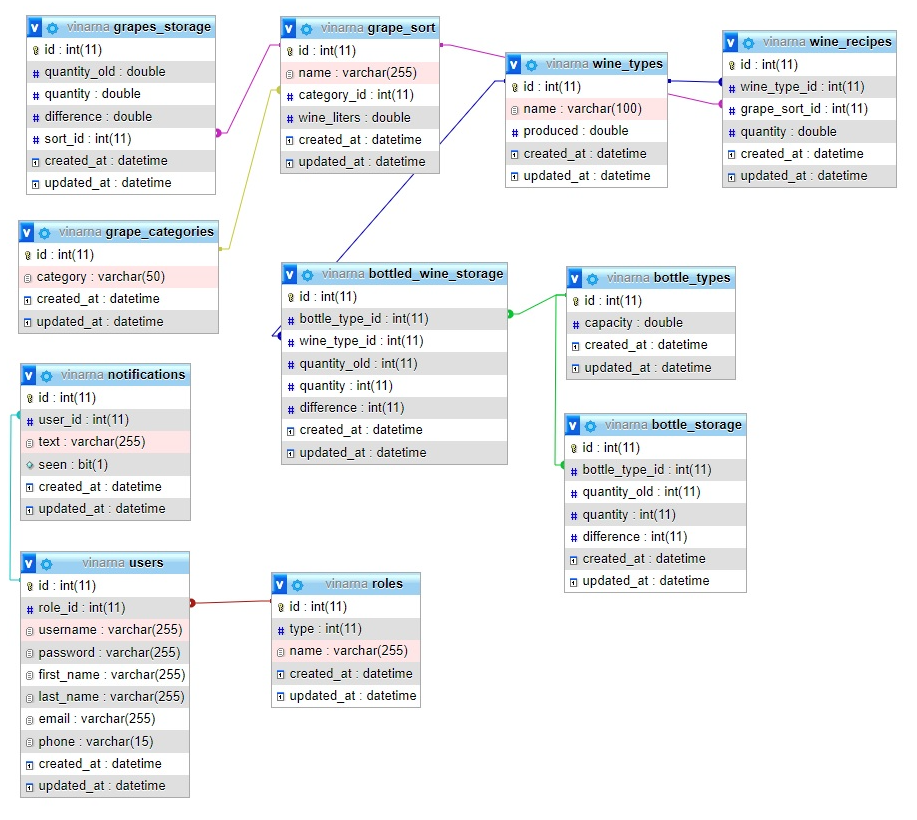


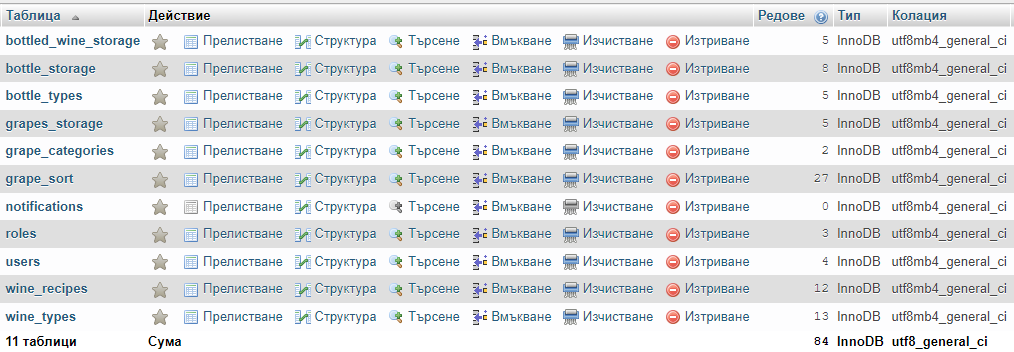
Последната таблица е bottled\_wine\_storage и отговаря на бутилираните вина в системата. Има връзки към bottle\_types и wine\_types с цел гъвкавост и за да може ефективно да се укаже съхранените бутилирани вина в склада точно в какъв тип бутилка и от какъв тип вино са.



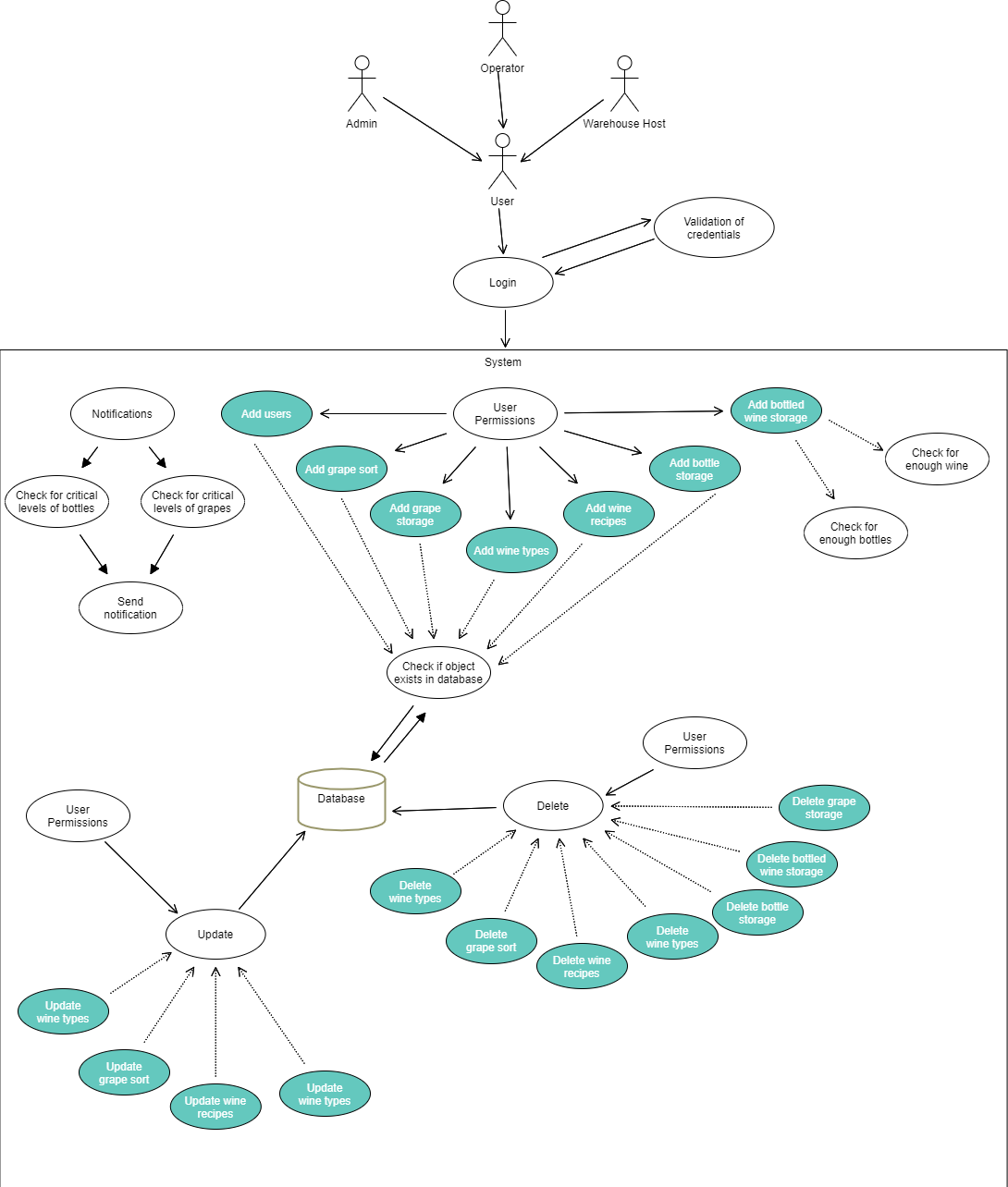
Във всички таблици в системата фигурират две datetime полета в края - created\_at и updated\_at. Това е добра практика при проектиране на почти всяка система. Целта е да може лесно да се проследи точно кога са създадени всички записи в базата от данни и за извличане на данни при нужда от актуални справки.

Това представлява базата от данни в пълния и вид, както и описаните релации:

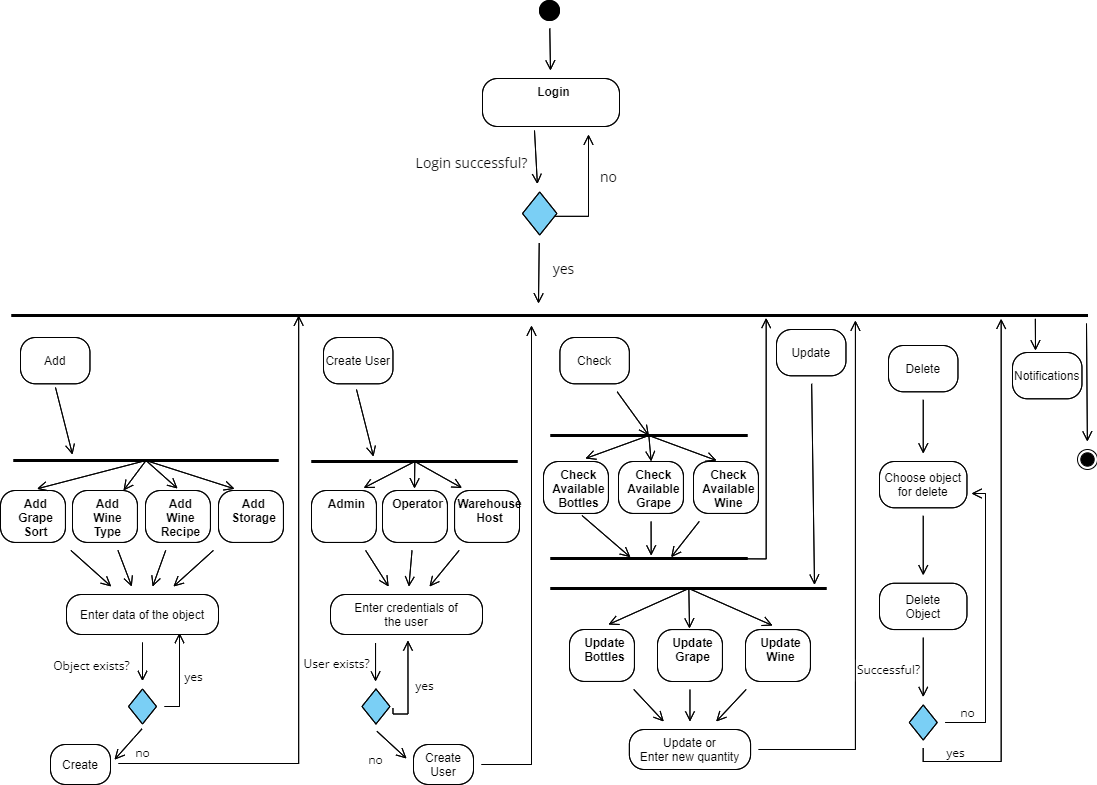




**Use-case диаграма:**



**Activity диаграма:**

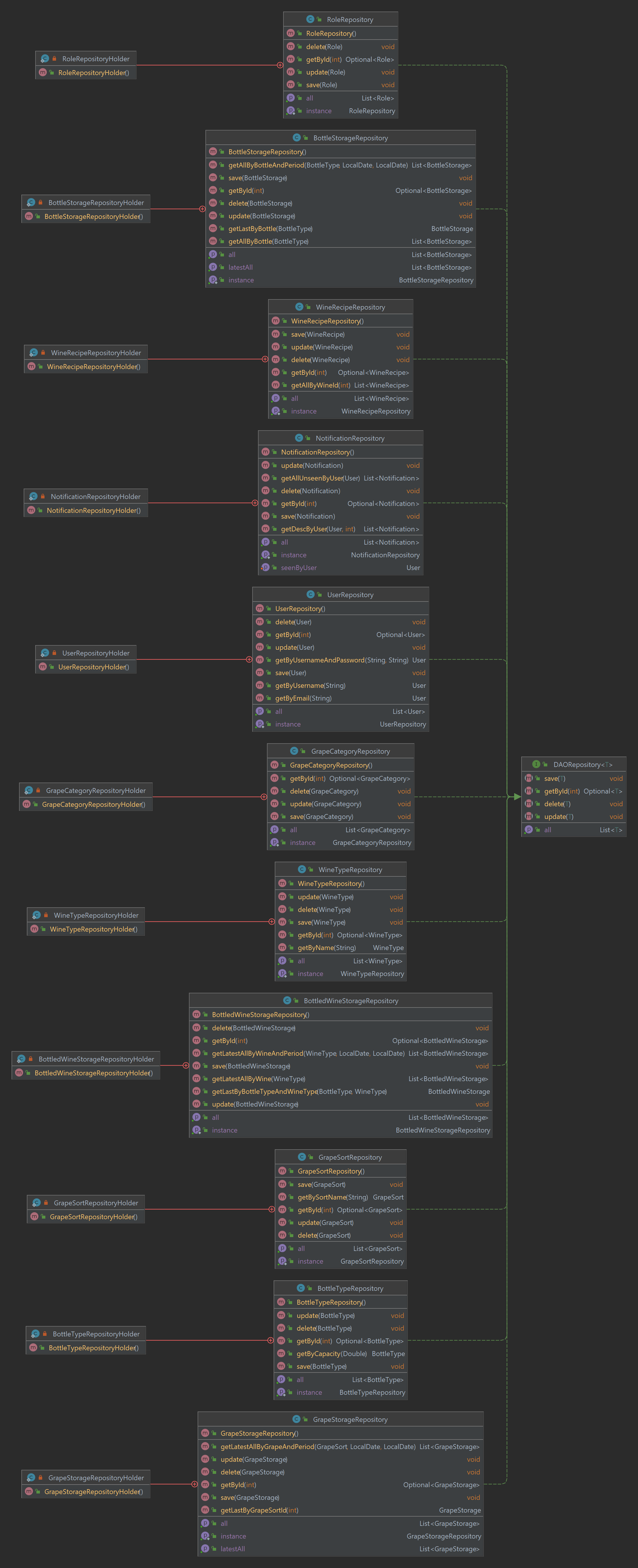


**Class диаграми:**

**- Entities**

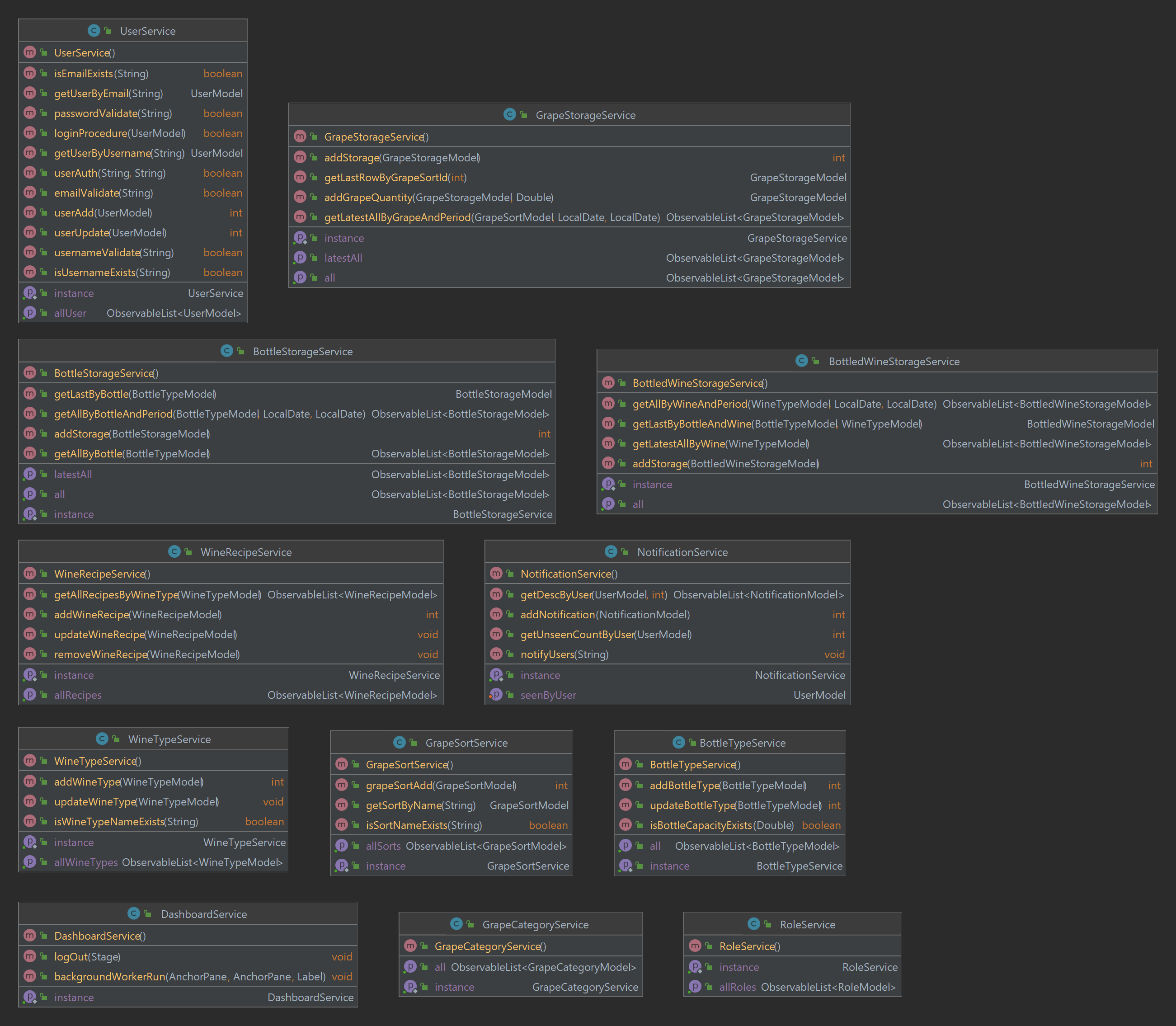
  
  
За пълен размер на Entities диаграмата: <https://imgur.com/YfAfNMq>

**- Repositories**



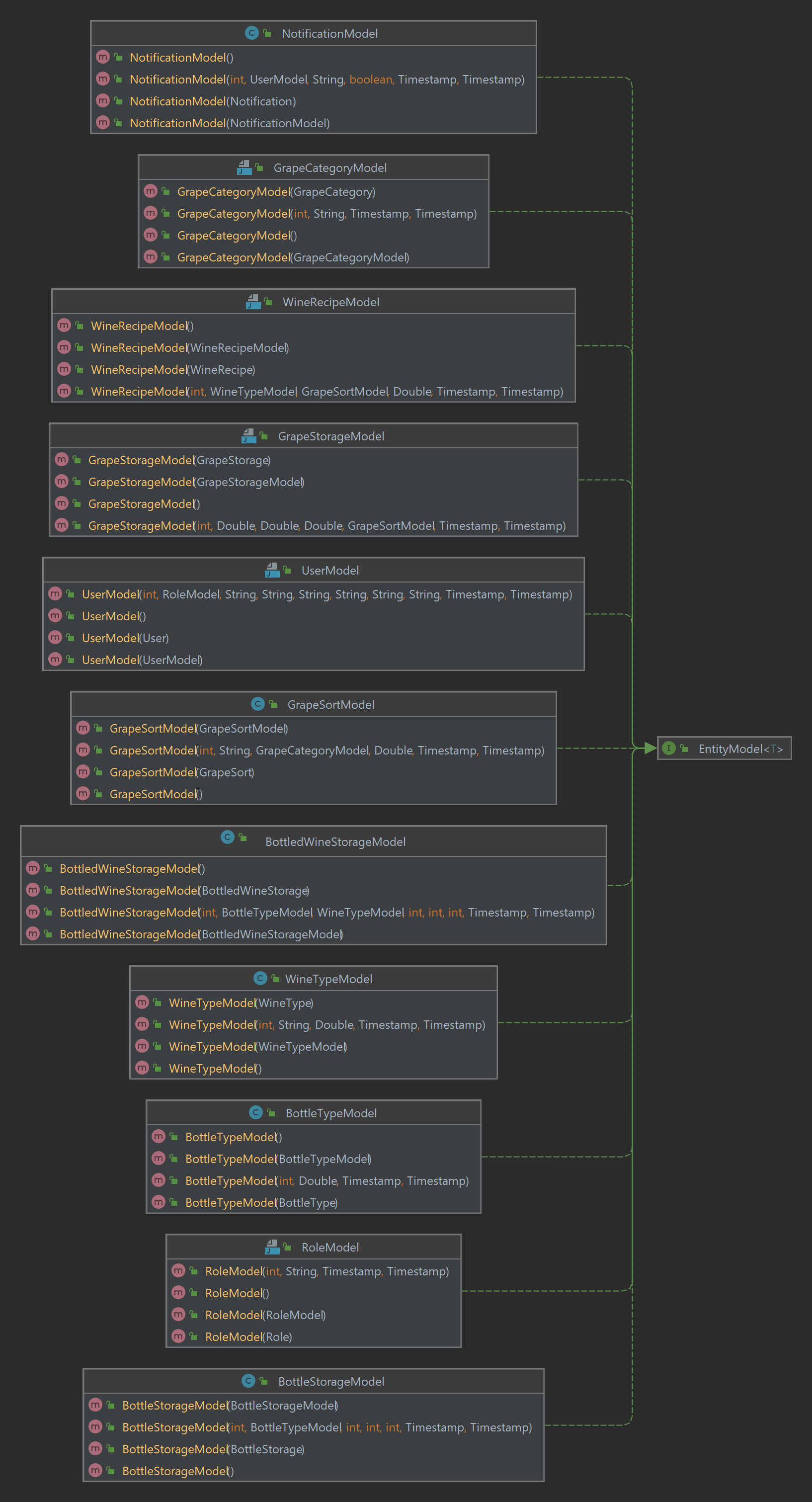
Снимката на диаграмата за Repositories в пълен размер може да бъде разгледана тук: <https://imgur.com/ZMVqInc>

**- Services**



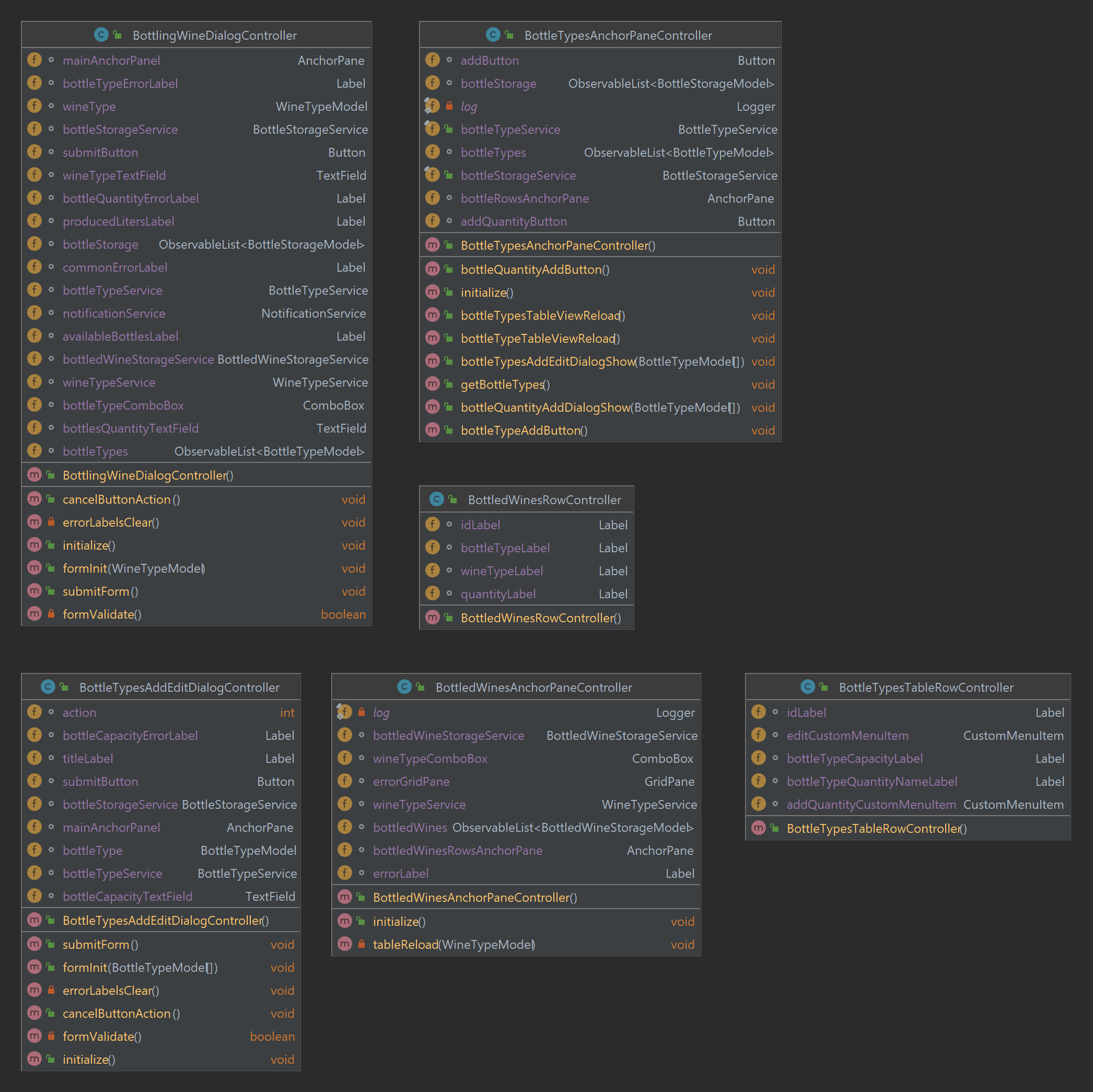
За пълен размер на Services диаграмата: <https://imgur.com/hUhyN0I>

**- Models**

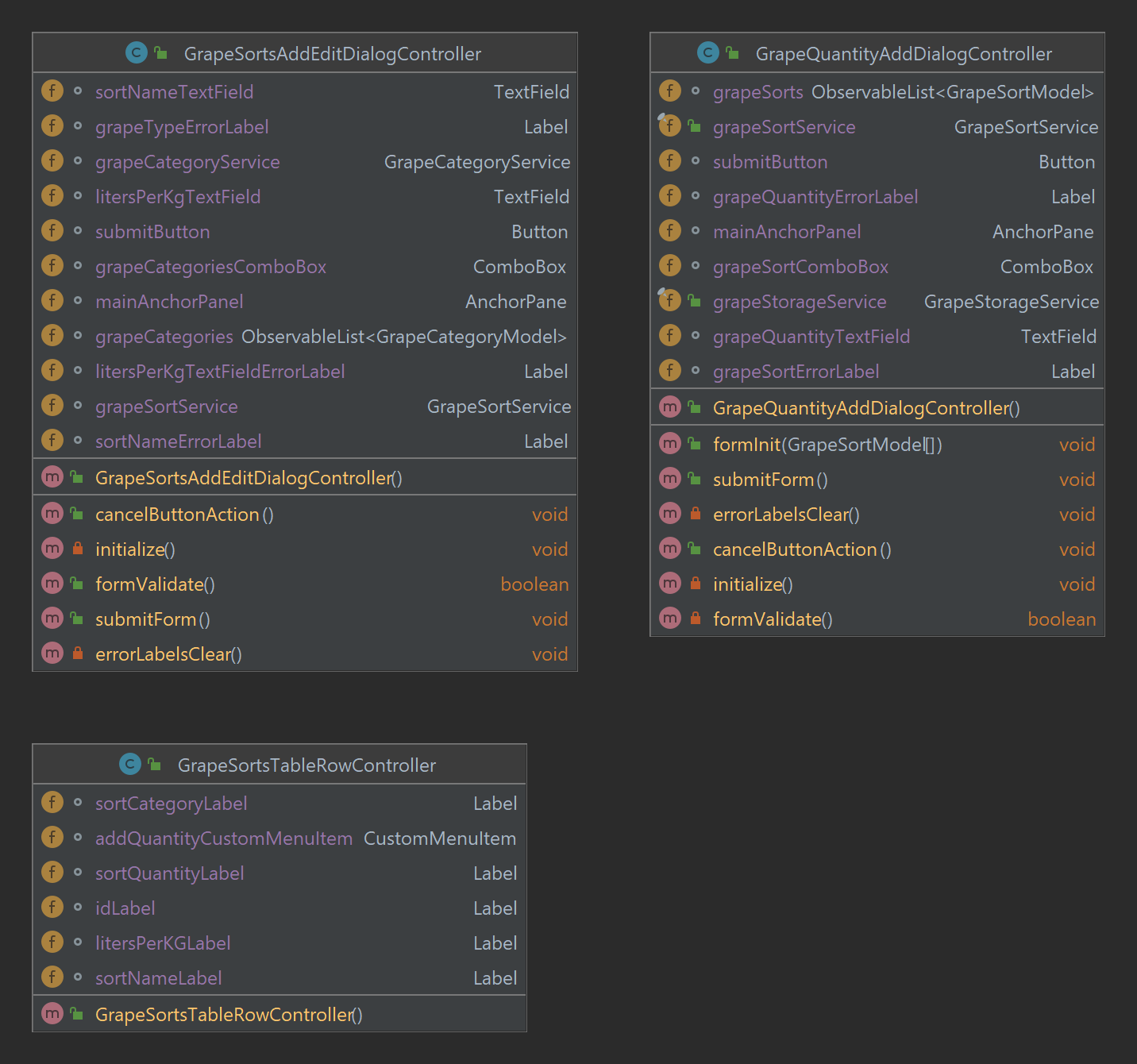


За пълен размер на Models диаграмата: <https://imgur.com/tAMdZpf>

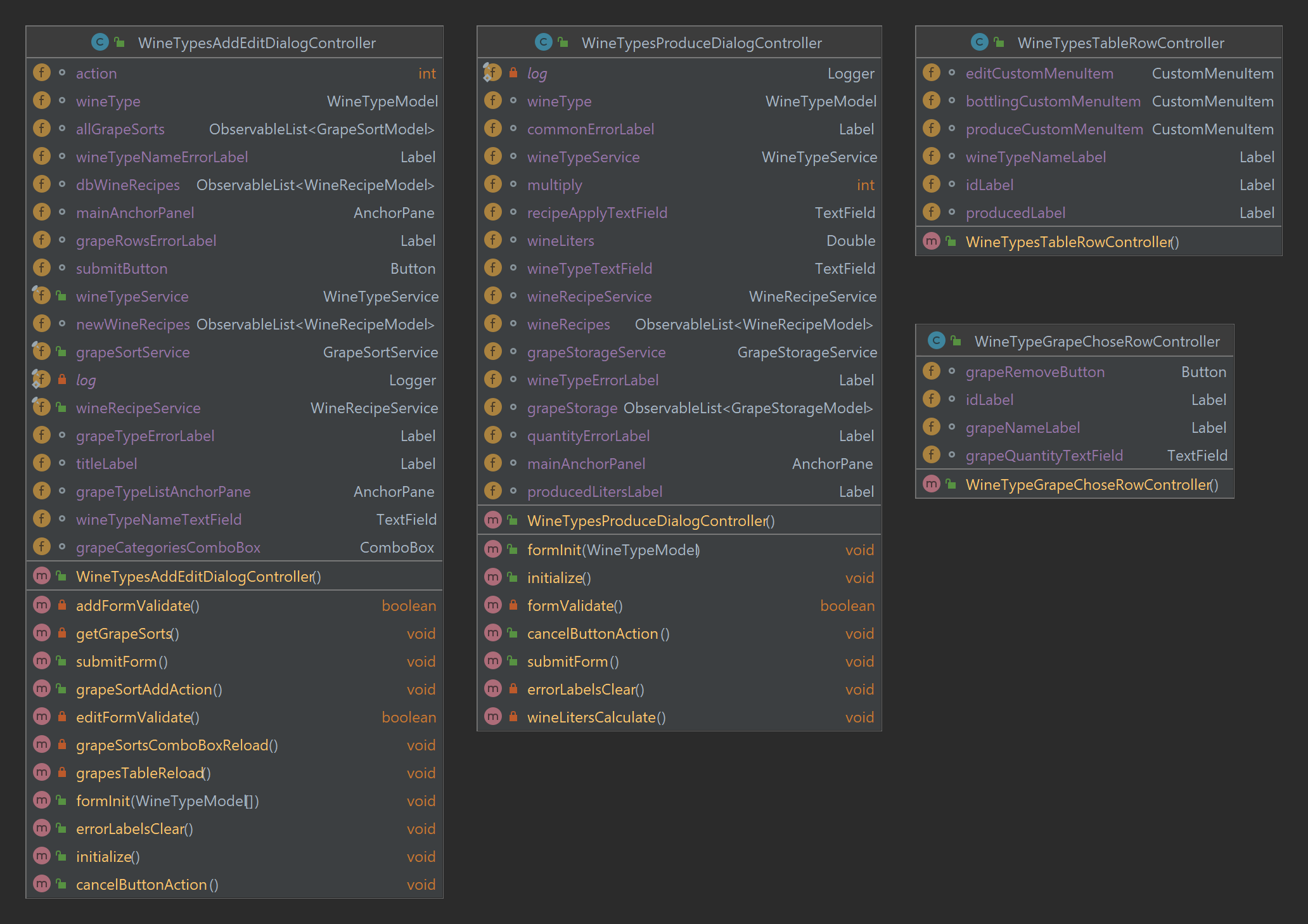
**- Bottle Controllers**



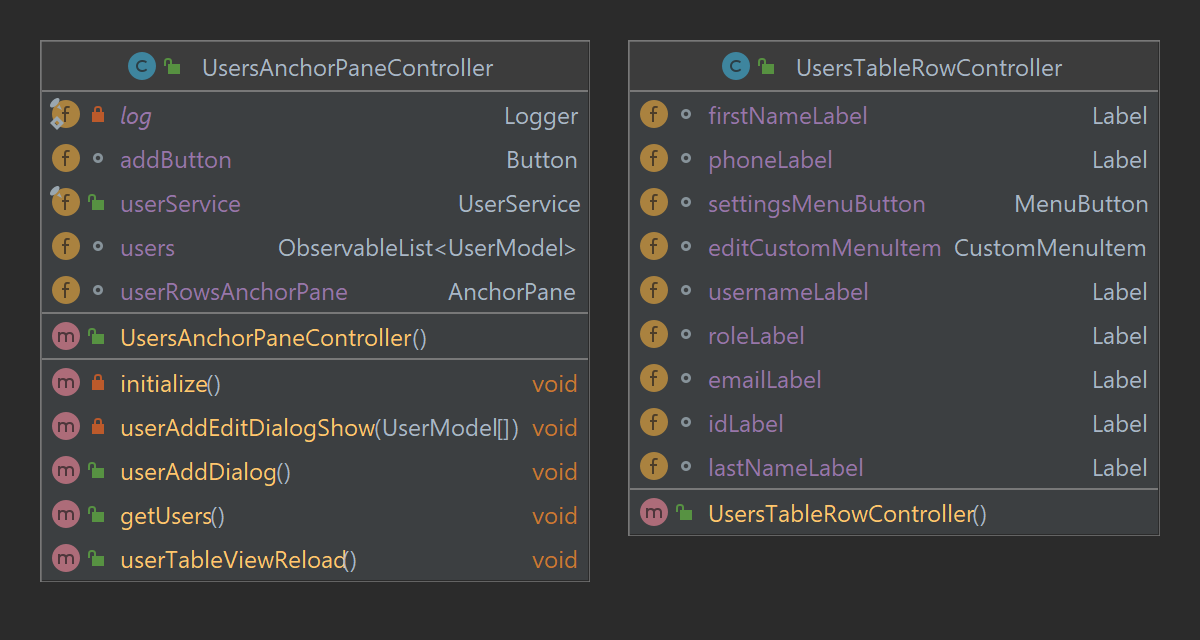
**- Grape Controllers**



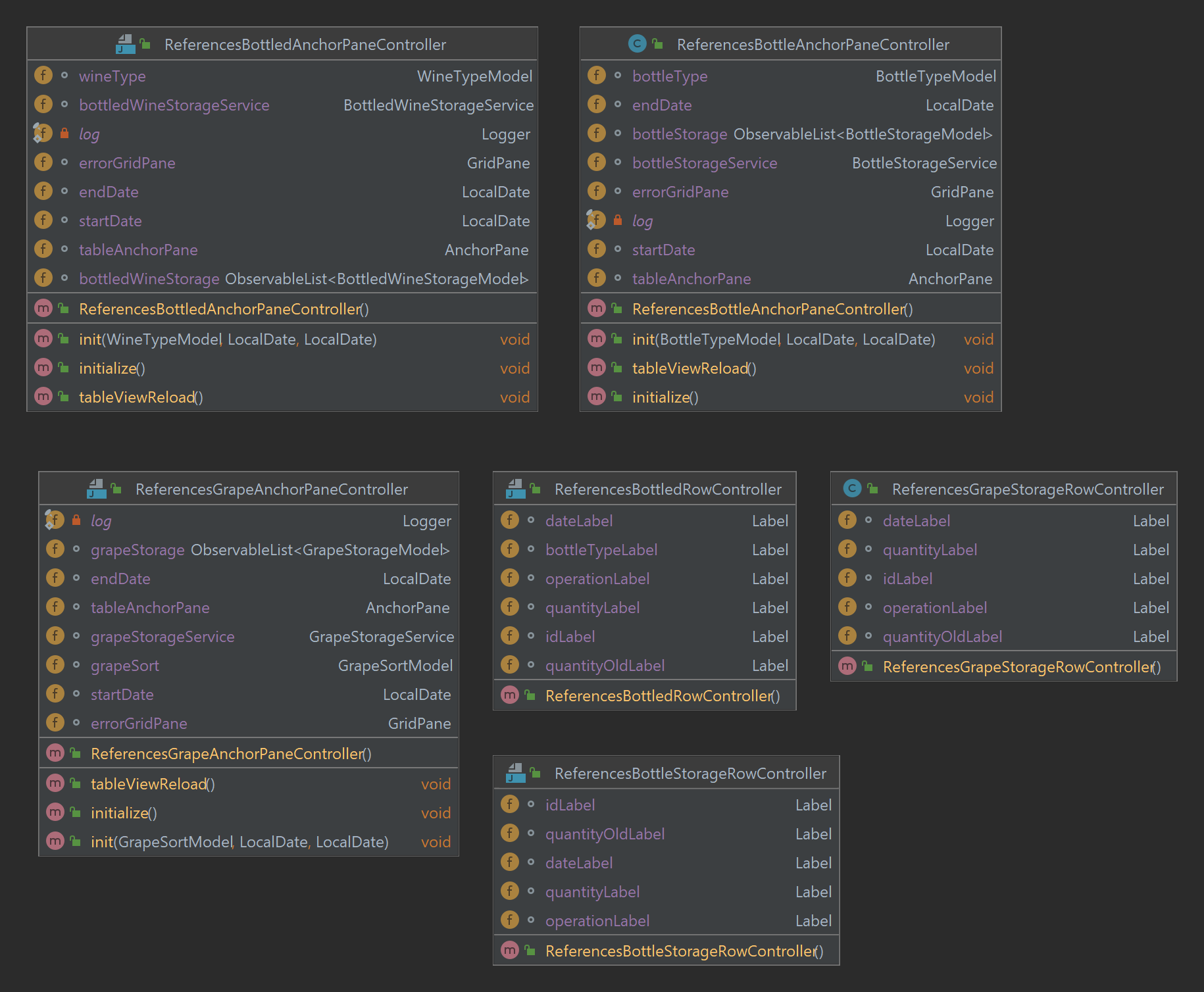
**- Wine Types Controllers**



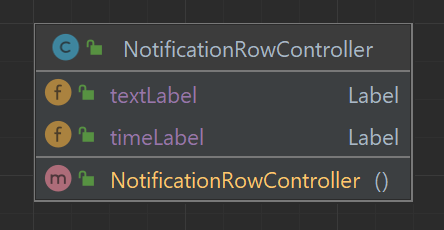
**- Users Controllers**



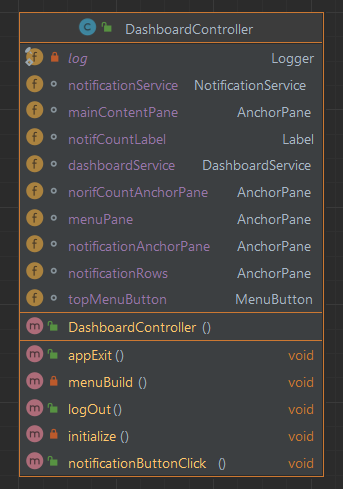
**- References Controllers**



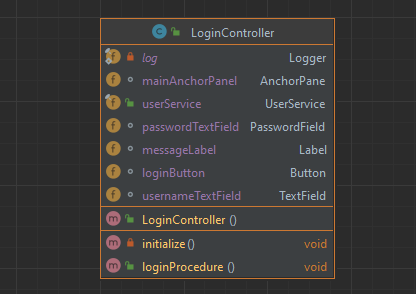
**- Notification Row Controller**



**- Dashboard Controller**

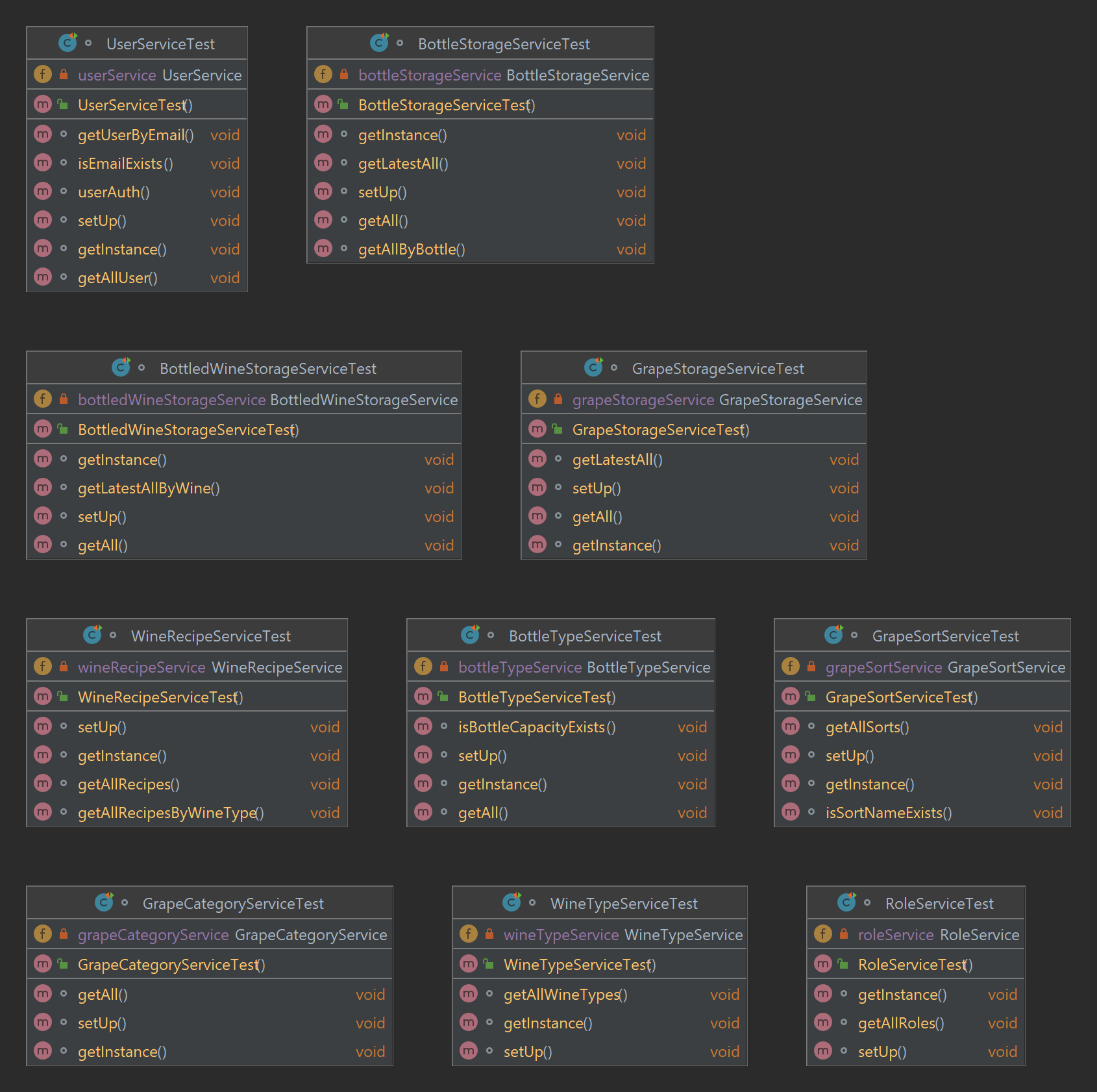


**- Login Controller**



**JUnit тестове:**

**- Business Tests**



**- Models Tests**

