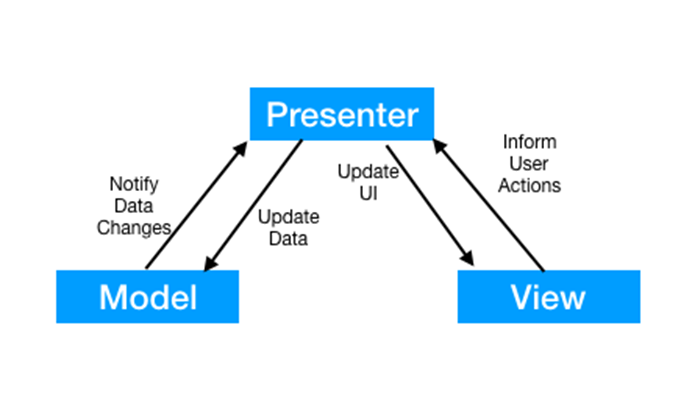
# Архитектурен Проект

## Основни цели на архитектурата

За приложението “EduWatch” е избран Model–view–presenter модел на архитектура. Той е разновидност на Model–View -Controller (MVC) модел. MVP е удачен за използване при създаването на потребителски интерфейси. Архитектурният стил разделя приложението на три градивни основни блока: Model, View, Presenter. Важна част от изграждането им е те да бъдат максимално разделени един от друг, за да не се нарушават принципите на архитектурата.

Всеки един от структурните блокове има строго определени функции. Програмата работи с потребителски интерфейс, който ще бъде свързан с другите градивни части на архитектурата.



### **Model:**

* Слой за съхранение на данни
* Отговаря за комуникацията с базата данни
* Осведомява Presenter при извличане , актуализиране и изтриване на данни.
* Presenter извлича от Model данните за визуализиране в потребителския изглед.
* Model и View нямат пряка връзка помежду си /взаимно не знаят за съществуването си/.

### **Presenter:**

* Слой съдържащ логиката на приложението „EduWatch“.
* Предназначението му е да взима данните от Model и да прилага логиката на потребителския интерфейс /решава какво UI да покаже/.
* Управлява състоянието на изгледа /View/ и предприема действия според известието за въвеждане на потребителя от View.
* Основен компонент за обмен на данни между Model и View.

### **View:**

* Слой на потребителския интерфейс.
* Осигурява визуализация на данните.
* Проследява действията на потребителя , за да осведоми Presenter.

## **Особености на архитектурата**

View и Model не си взаимодействат директно. Това подобрява имплементацията на View.

Висок риск крие Presenter. Той може да се разшири до огромен всезнаещ клас , което води до високо ниво на сложност.

Важно е да се следва последователността при изготвянето на софтуерния продукт, за да се избегнат възможните проблеми и за да може крайният продукт да е висококачествен и максимално оптимизиран.

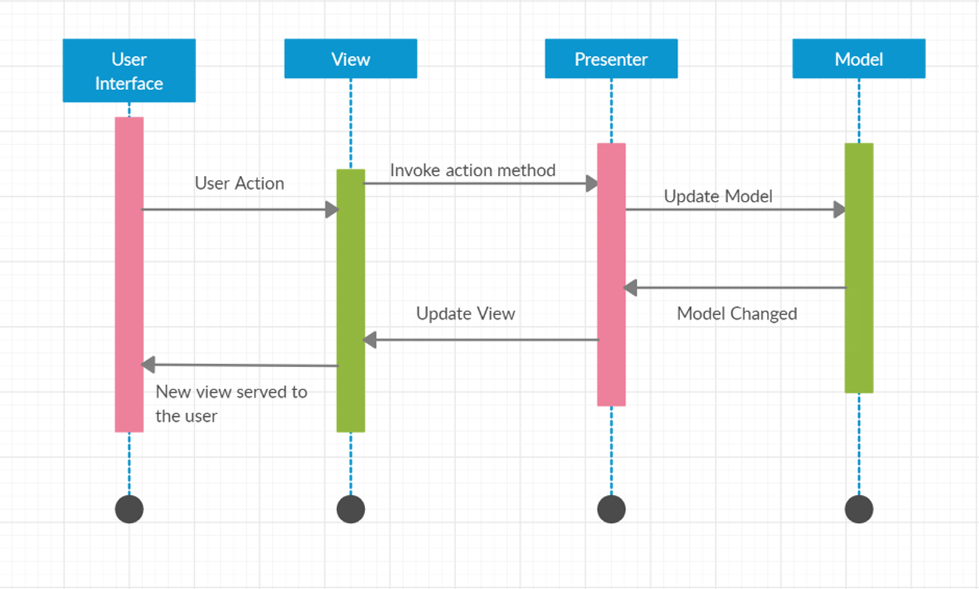
## **MVP и Windows Forms**

Защо е избрана архитектурата:

Основен проблем при използването на МVC-базирани модели е използването на потребителския интерфейс. Често се наблюдава, че той извършва повече функционалности отколкото би трябвало да са му възложени. Липсата на ясно разпределение между модулите довежда до това, че потребителският интерфейс изземва голяма част от логика принадлежаща на останалите слоеве.

MVP решава тази често допускана грешка. Неговата цел е да остави на потребителския интерфейс само кода и логиката, които са му необходими. По този начин се постига изграждането на рационализиран и защитен код за многократна употреба.

# 4.Логически изгледи

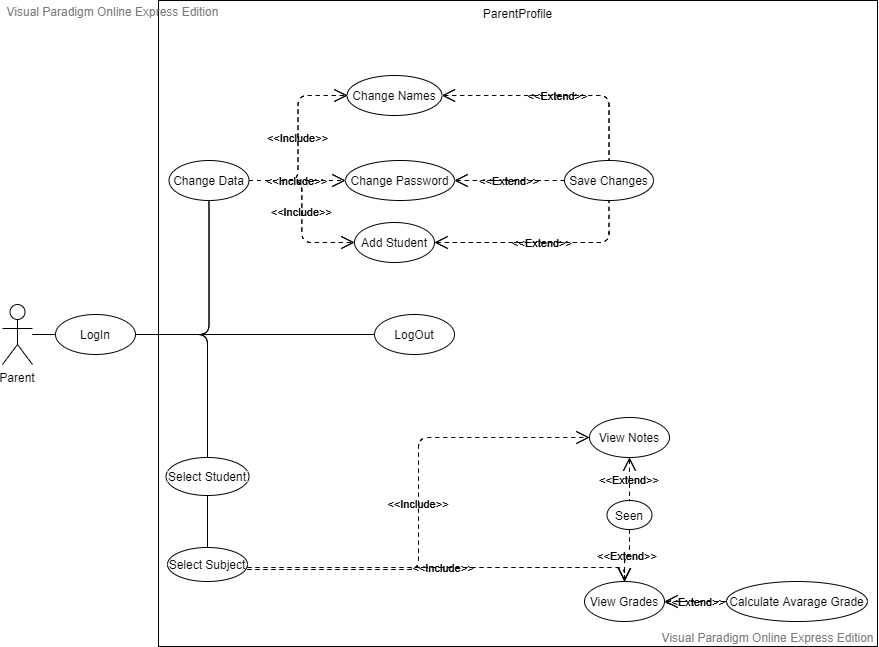


Следната Sequence диаграма показва работата между отделните компоненти на приложението. Потребителския интерфейс праща заявка към View. View показва, че е извикан метод за действие от Presenter. Работата на Presenter е да се извлече нужната информация от Model. Model получава заявката за извличане на данни от Presenter и му връща нужната информация, която Presenter препраща към View. View актуализира новия изглед към потребителския интерфейс. Диаграмата демонстрира работата и отношенията на всеки един модул от MVP архитектурата, без излишни стъпки и с точно разпределение на всеки един случващ се процес. Основните три модула са скрити от обикновения ползвател, така че достъпът да му до данни, несъществени за неговата работа, да не бъде осъществим. Това е основна цел за приложенията свързани с масово или неспециализирано ползване –потребителя получава само услугата, от която има нужда.

##### 4.1. Сценарии на употреба на приложението „Eduwatch“.

В приложението съществуват четири случаи на ползване:

* Първият случай е при използване от потребител от тип родител.

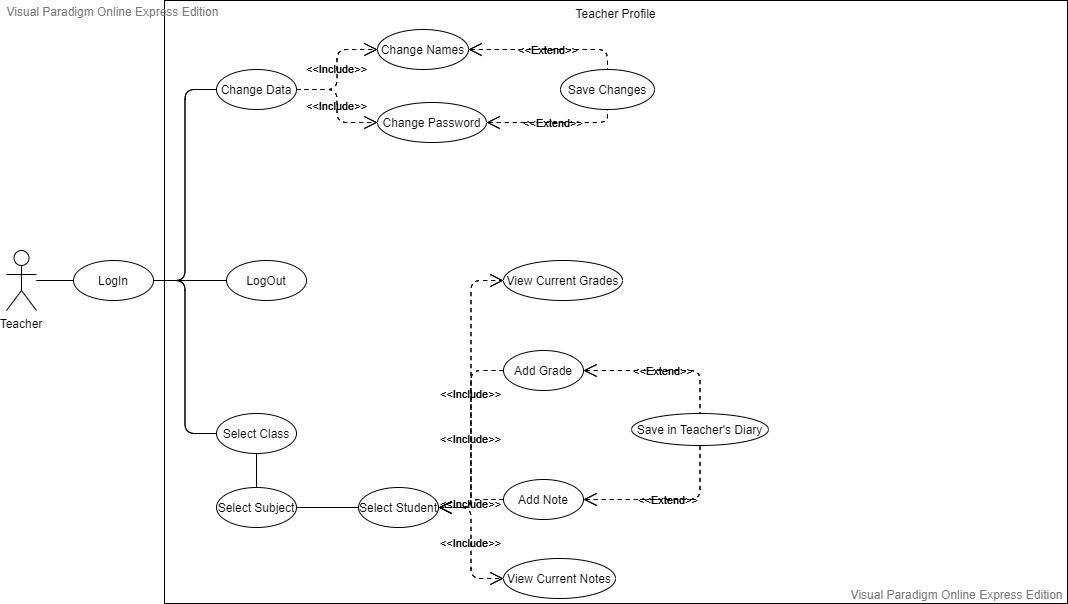


Следната Use-case диаграма показва вариантите за действия на родител в системата. Лицето се вписва успешно и се препраща към своя персонализиран профил. Той има няколко опции. Една от тях е да редактира своята информация , което включва да промени името и фамилията си /Change Names/, да промени паролата си/Change Password/ и да асоциира ученик към себе си . Всички тези алтернативи за промени , потребителя може да реши дали да запази или не.

Втория сценарии към родителския профил включва проследяване на информацията за асоциираното с него дете. Съответно потребителя избира ученика си , след което и учебна дисциплина. Ползвателя има възможност да види текущите оценки и забележки по посочения предмет, както и да отбележи , че е видял текущите данни. При избор на оценки , той може да пресметне и средния успех на базата на досегашното оценяване.

Третия вариант е напускане на родителския профил и връщане към вписване в системата .

* Втория случай е при използване на приложението от потребител тип – учител.

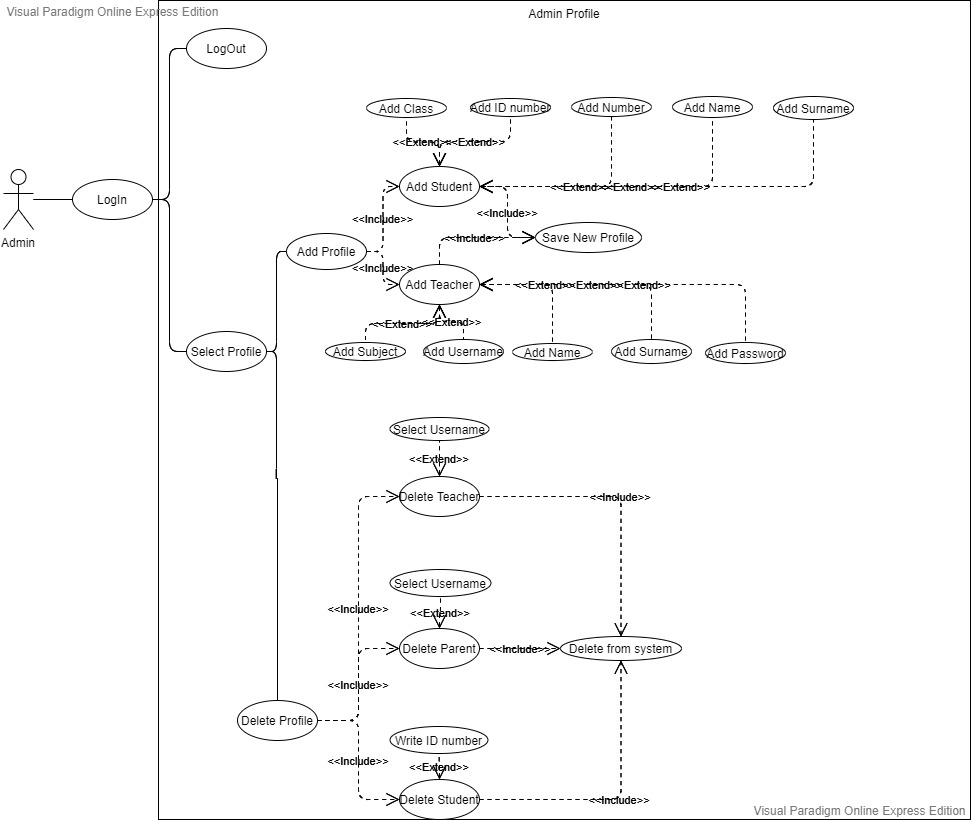


На втората Use-case диаграма са посочени сценариите за употреба на учителя. След успешно вписване в системата лицето се препраща към личния си учителски профил. Съществуват няколко варианта на действия. Един от тях е редакция на текущите му данни , която включва промяна на име и фамилия/Change Names/ и промяна на парола/Change Password/. Потребителя може да реши дали да запази настъпилите модификации.

Втория вариант за дейност е свързан с учителския дневник. Потребителя избира клас , предмет и ученик. Той може както да провери текущите оценки и забележки , така и да добави и запази нови такива.

Третия вариант е напускане на учителския профил , което връща лицето към вписване в системата.

* Третия случай е при използване на приложението от потребител тип – админ.

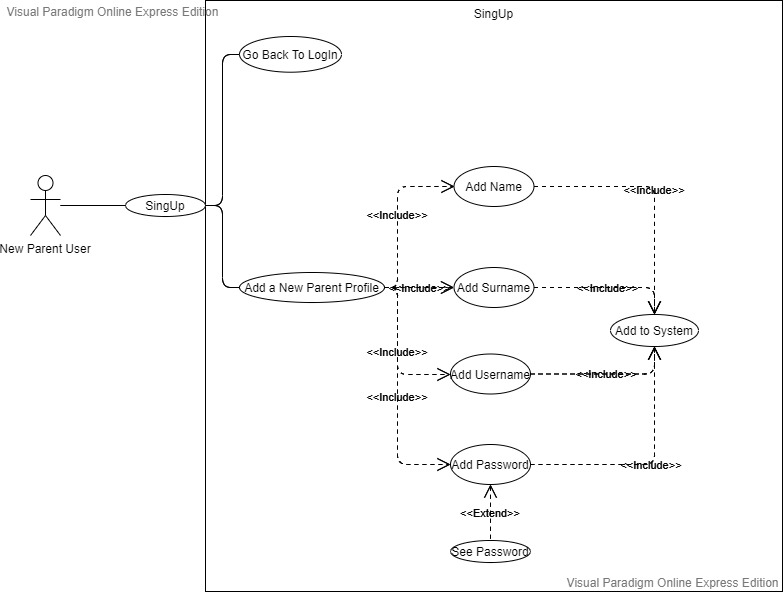


Третата Use-case диаграма показва сценарий за администратора. Потребителя се вписва в успешно системата. В този профил ползвателя има две основни дейности добавяне и изтриване на другите ползватели на приложението. Потребителя избира за кого желае да направи промяна. След това решава каква да бъде тя /изтриване или добавяне/. Ако определи добавяне на лице , администратора за случай на ученик трябва да въведе клас ,ЕГН , номер, име и фамилия. Ако е избрано добавяне на учител , извършителя на дейността трябва да въведе предмет , потребителско име , име, фамилия , парола. Администратора запазва настъпилите промени.

В случай , че ползвателя реши да изтрие съществуващ профил следва – при случай на учител се избира потребителското име , което трябва да бъде премахнато; при случай на родител отново заличаването се случва , чрез избор на потребителско име ; при вариант ученик се въвежда ЕГН на ненужното лице. Администратора решава дали да запази настъпилите промени.

Друг сценарии на профила е напускането на системата , то може да се случи независимо дали са направени промени. При напускане на администраторския режим , системата връща ползвателя към вписване.

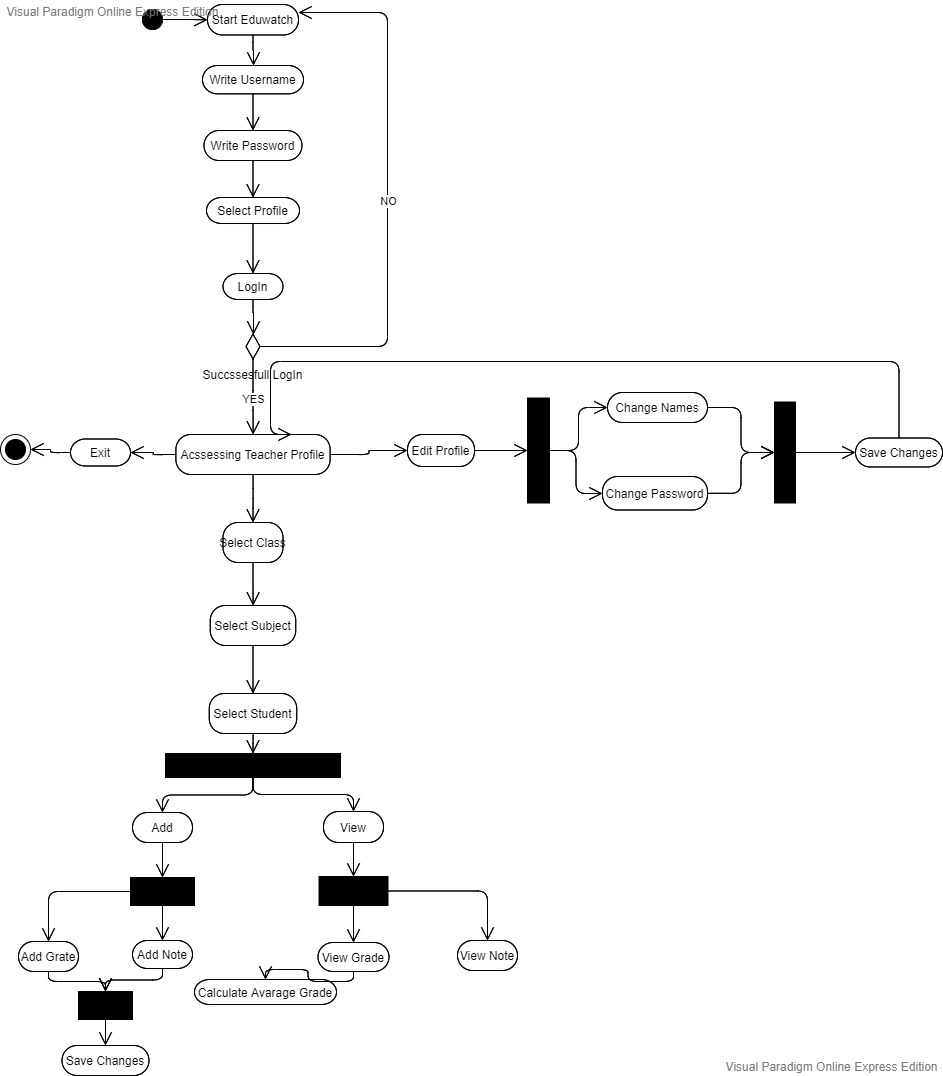
* Четвъртия случай на използване на приложението е при нов потребител , който желае да създаде свой родителски профил:



Диаграмата показва вариантите на нов потребител при вписване в системата. Родителя решава да създаде нов профил. Ако той вече има такъв , съществува вариант , в който той може да се върне обратно към вписването в системата. Ако няма създаден профил , ползвателя трябва да въведе данните си – име , фамилия , потребителско име , парола с вариант за виждането и .След успешното попълване родителя се добавя към системата и вече може да я достъпва като останалите съществуващи родителски актьори.

## 4.2.Диаграми на активността/действията.

В няколко Activity диаграми са изобразени моделите на бизнес процесите в приложението “Eduwatch”. В случая те са направени паралелно.

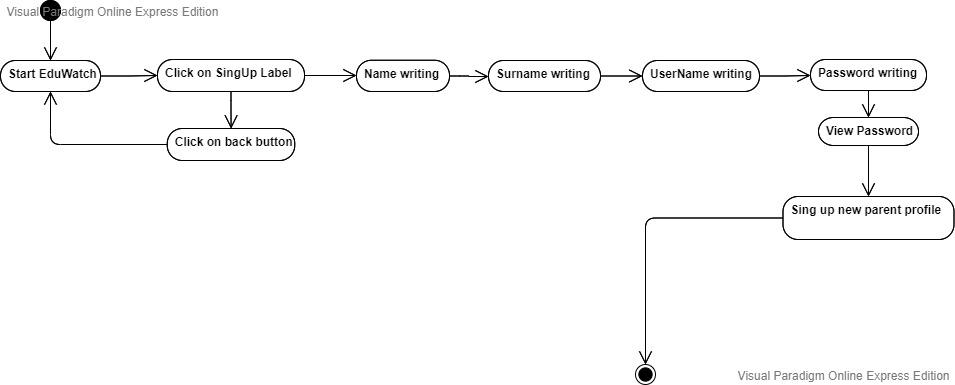
* Първата от тях се отнася към дейностите извършвани в учителския профил

Потребителя стартира приложението. Трябва да се въведат потребителско име , парола и да се избере вида профил , който ползвателя иска да достъпи. Ако вписването е неуспешно , системата връща лицето за ново въвеждане. При успешна автентикация лицето влиза в персонализирания си учителски профил. В него ползвателя има възможност да избере клас , предмет и ученик от паралелката. След това е създадена опцията да се видят текущи оценки и забележки или да се добавят нови такива.

Друга дейност в учителския профил е негова настойка. Това е промяна на име , фамилия и парола и запазването им.

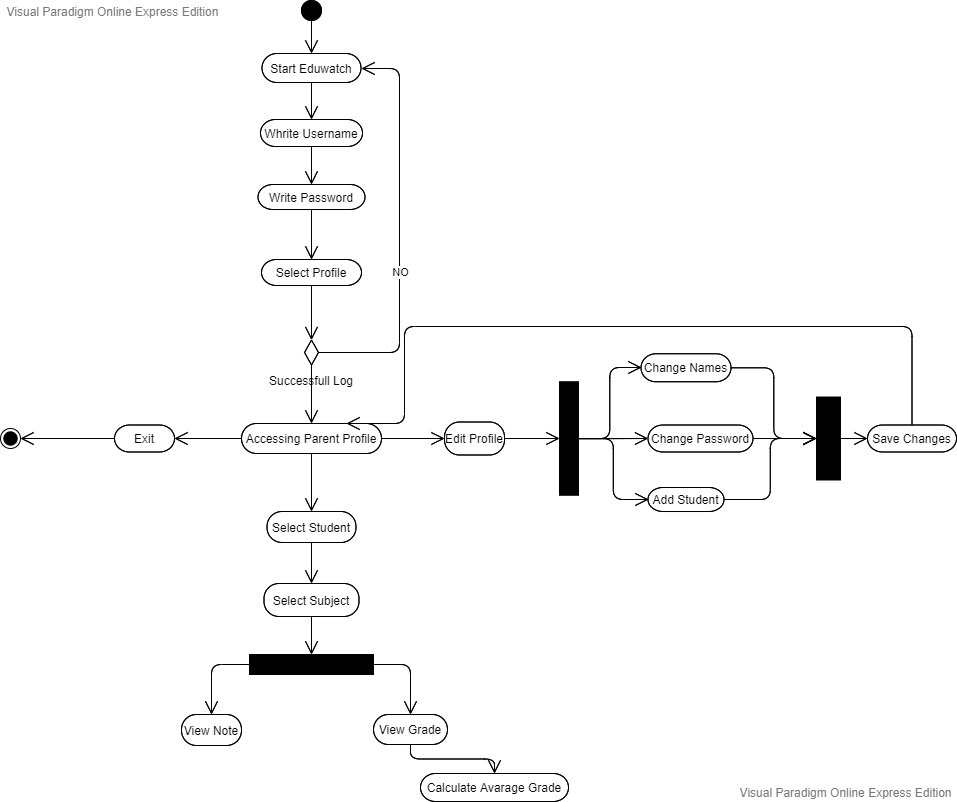
Третата активност е изход от профила. Тя е възможно да се случи във всеки един момент и е независима от останалите.

* Втората диаграма се отнася до дейности при създаване на нов родителски профил:



Потребителя стартира приложението и решава да създаде нов родителски профил. Ако има вече такъв се връща в началното вписване в системата. За генериране на ново лице е нужно въвеждане на име, фамилия , потребителско име и парола , която може да се види. След което родителя създава новия си профил и вече може да се впише в системата.

* Следващата диаграма се отнася за дейностите извършвани в родителския профил :

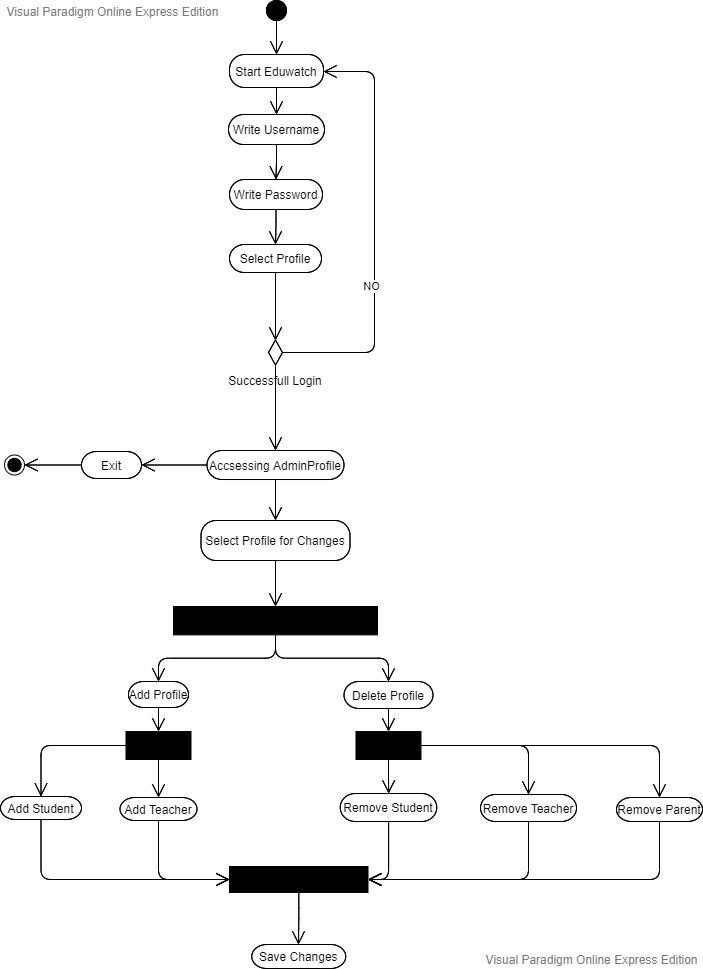


Потребителя стартира приложението. Трябва да се въведат потребителско име , парола и да се избере вида профил , който ползвателя иска да достъпи. Ако вписването е неуспешно , системата връща лицето за ново въвеждане. При успешна автентикация лицето влиза в персонализирания си родителски профил. В него потребителя има възможност да избере ученик и предмет , след което да види текущите оценки и забележки. При избор за визуализиране на текущи оценки съществува опция за изчисляване на среден успех.

Друга дейност за родителския профил е промяна и запазване на данни – име , фамилия , парола , добавяне на ученик. При неуспешно добавяне системата връща потребителя за ново въвеждане.

Третата активност е изход от профила. Тя е възможно да се случи във всеки един момент и е независима от останалите.

* Четвъртата диаграма показва последователността от активности в администраторския профил:



Потребителя стартира приложението. Трябва да се въведат потребителско име , парола и да се избере вида профил , който ползвателя иска да достъпи. Ако вписването е неуспешно , системата връща лицето за ново въвеждане. При успешна автентикация лицето влиза в администраторския профил. Ползвателя избира за кое лице иска да направи промени – учител , родител , ученик.Промените са два типа добавяне на профил /важи за ученик и учител/ и изтриване на профил/важи за всички/. Админът запазва настъпилите промени.

Друга активност е изход от профила. Тя е възможно да се случи във всеки един момент и е независима от останалите.

## 4.3. Клас-диаграма

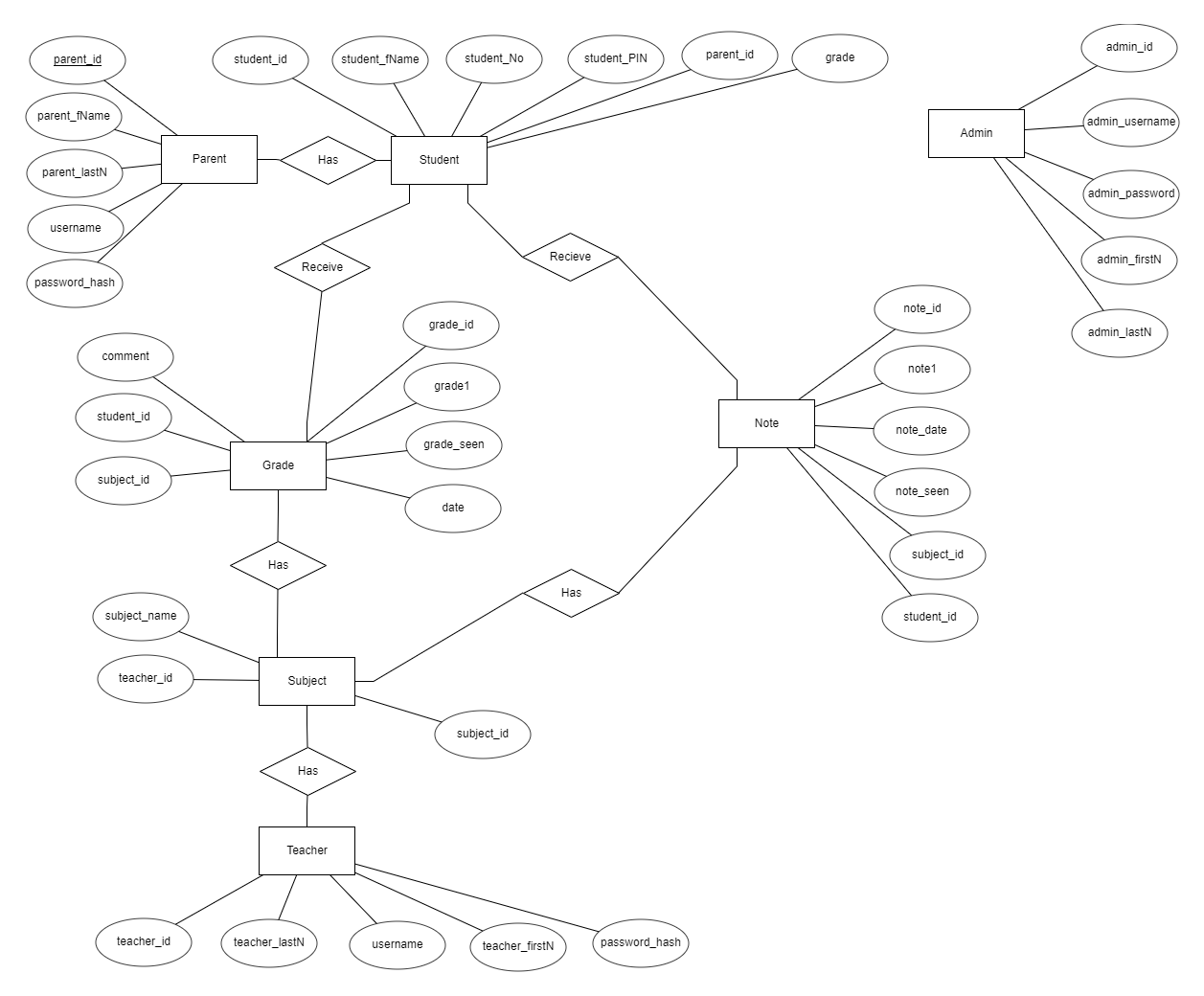
В клас-диаграмата са показани отделните модули на приложението , заедно с техните класове и зависимости помежду им .

[Линк към диаграмата](MVC%20Structure.vpd2%20(1).png)

## 4.1.Entity Relationship диаграма

В този тип диаграма е отразена базата данни. Тя показва нейните обекти , техните свойства и взаимодействията между тях. В случая обектите са родител , ученик , учител , администратор , предмет , оценка , забележка. За всеки един от тях са изобразени съответните им свойства. Връзките помежду им са следните:

* Всеки родител е свързан ученик.
* Всеки ученик получава забележка и оценка.
* Всеки предмет има забележка и оценка.
* Всеки учител преподава по някой предмет.
* Администратора не е свързан с останалите обекти от базата данни.



# 5.Нефункционални изисквания

1.Използваемост /Usability/

Програмата е създадена , така че да е удобна за използване дори от потребители с невисоко ниво на компютърна грамотност. Не се изискват специални знания за да се използва системата. Добавянето на нови функционалности към “EduWatch” не биха затруднили досегашните ползватели. Чрез интуитивния потребителски интерфейс , лесно и бързо се изпълняват основните задачи при първата си среща с него. При допускане на грешки системата реагира и лицето лесно може да ги поправи.

2.Разширяемост/Extensibility/

В “Eduwatch” е предвидено бъдещо развитие , без да се нарушават системни функции. Могат да бъдат нови опции.

3.Сигурност/Security/

Системата осигурява защита на личните данни , като използва Salting – криптографски метод за добавяне на допълнителни символи към хеша на паролите , което ги прави устойчиви на атаки от твърди сили. Потребителите могат да достъпват единствено персонализираните си профили след въвеждане на коректни потребителско име и парола.

4.Достъпност

Приложението не използва интернет връзка и може да се използва на всяко устройство след успешна инсталация. Могат да възникнат проблеми с достъпа до профилите , ако се наруши в даден момент връзката с базата данни.

5.Възможност за тестване

Потребителският интерфейс предоставя лесно ръчно тестване. В зависимост от свързаността между отделните компоненти могат да се използват Unit или Integration тестове.