**ПРЕДМЕТ: РАЗРАБОТКА НА СОФТУЕР**

**ДЕЙНОСТ: УЧЕНИЧЕСКИ ПРАКТИКИ 2**

***2021-2022***

**Тема: „Информационна уеб базирана система “Кошер за изкуствен генно модифициран вид пчели”“**

**Курсов проект**

*Автор:*

*Тодор Динков Божинов 11Б*

*Ангел Веселинов Желязков 11Б*

БургасСъдържание

[1 Въведение 2](#_Toc648282184)

[2 Цели и обхват на софтуерното приложение 3](#_Toc1024658360)

[3 Анализ на решението 3](#_Toc40238208)

[3.1 Потребителски изисквания и работен процес 4](#_Toc378397095)

[3.2 Примерен потребителски интерфейс 5](#_Toc1745326166)

[3.3 Диаграми на анализа 6](#_Toc632185600)

[3.4 Модел на съдържанието / данните 6](#_Toc1444790647)

[4 Дизайн 6](#_Toc48057846)

[4.1 Реализация на структура на приложението (3-layer), Разделение на кода според предназначението му 6](#_Toc1315351454)

[4.2 Организация и код на заявките към база от данни 6](#_Toc1353309475)

[4.3 Наличие и интуитивност на потребителски интерфейс (конзолен, графичен, уеб) 6](#_Toc1367663312)

[5 Тестване 6](#_Toc2056882221)

[6 Заключение и възможно бъдещо развитие 6](#_Toc210501827)

[7 Използвани литературни източници и Уеб сайтове 6](#_Toc942551430)

[8 Приложения 6](#_Toc845044098)

[9 Критерии и показатели за оценяване 6](#_Toc1819287021)

# Въведение

Настоящият документ представлява шаблон за оформление на документация на курсов проект за предмета „Разработка на софтуер“.

Проекта засяга пряко проблема свързан с пчелите. Той е доста актуален днес и ще става все по и по актуален.

Пчелите и другите насекоми опрашители имат ключов принос за поддържането на екосистемите и биоразнообразието. Липсата на опрашители поставя под заплаха много растителни видове, а също и организмите, които са свързани пряко или непряко с тези растения. Много селскостопански култури също са подложени на риск. Затова този проект има за цел да разреши проблема, който е свързан с намаляването на пчелните видове.

В документацията ще видите какви са целите, анализ на решението, какъв ще е дизайнът, бъдещото развитие и използваните източници.

# Цели и обхват на софтуерното приложение

Идеята е да се създаде информационен сайт, който да дава максимална ясното по проекта за кошер за изкуствен генно модифициран вид пчели. Този кошер ще има за цел да създаде подходящите условия за размножаване и развитие на специфичния вид като това от своя страна ще има за цел максималната продуктивност на насекомите.

Като за начало в кошера ще бъде пусната вече оплодена женска пчела. След това започва процеса на снасяне, след 3дни ларвите се излюпват и чрез сензор се включва специален режим за хранене на новородените.

След този процес се очаква вече пчелите да са годни за опрашване на растения и производство на мед, а след това кошера започва с останалата част по поддържката на условията, пригодени към нуждите на насекомите.

Обаче, ако не е възможно да се създаде специалният вид пчела, ще бъде използван нормален такъв като тогава кошера ще бъде пригоден към характеристиките на обикновения вид, но отново ще има за цел да пресъздаде идеалните условия, така че ефективността и продуктивността да е максимална.

# Анализ на решението

## Потребителски изисквания и работен процес

Когато влезеш в сайта, първото което ще може да се види е началната страница, в нея ще има накратко информация свързана с другите менюта.

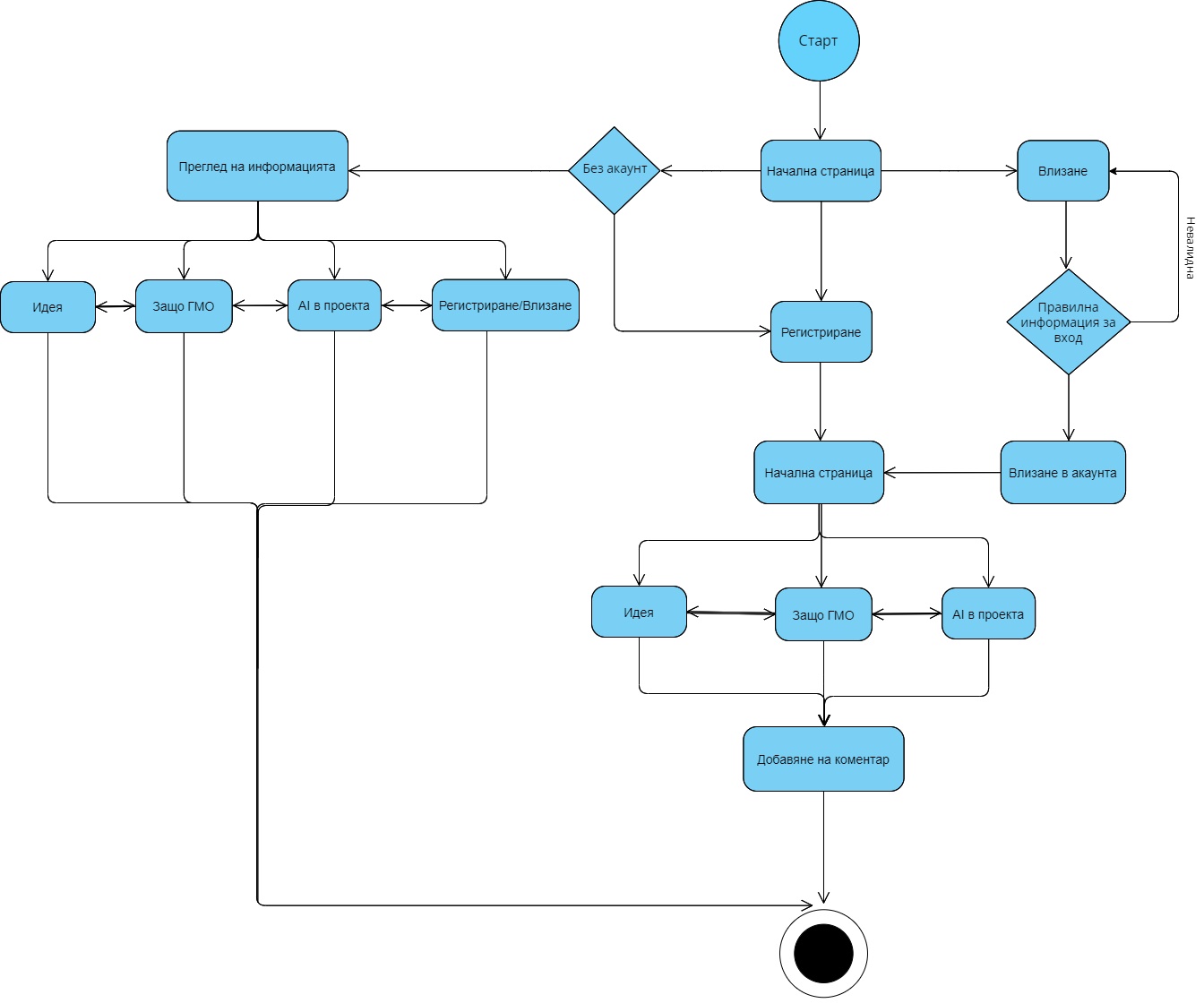
Съответно от началната страница ще има връзки към: Идея, Защо ГМО, АI в проекта, регистриране/влизане.

В менютата Идея, Защо ГМО и AI в проекта ще може да се прегледа информация в детайли за това как, защо и какво ще представлява този кошер и тези пчели.

Когато отидем в раздела за регистриране/влизане ще има две опции, да се регистрираш, ако вече нямаш акаунт или влезеш в своя такъв, като при въведени грешни данни за вход, ще бъдете върнати да въведете информацията си отново.

При успешна регистрация или влизане в акаунт ще можете да пишете коментари под статиите.

Приложената по-долу диаграма показва графично как работи сайта(*Фигура 3.1*)



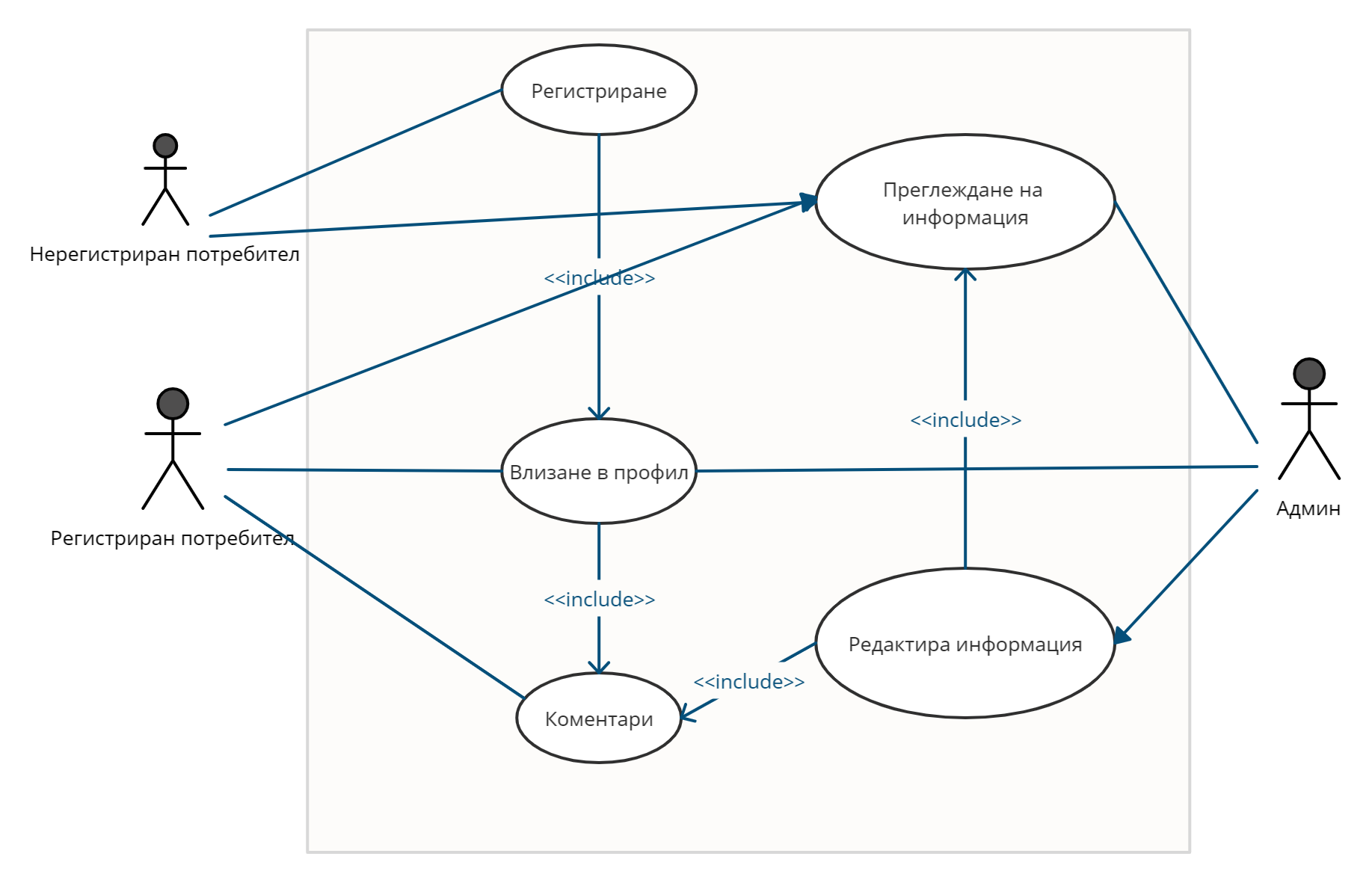
*Фигура 3.1*

Нерегистрираният потребител ще има избора: да се регистрира и влезе в акаунта си или да преглежда свободно информацията в сайта

Регистрираният потребител може да преглежда информацията, да влезе в акаунта си и после да напише коментари, ако желае.

Администратора ще може да редактира информацията, коментарите и данните за профилите при нужда.

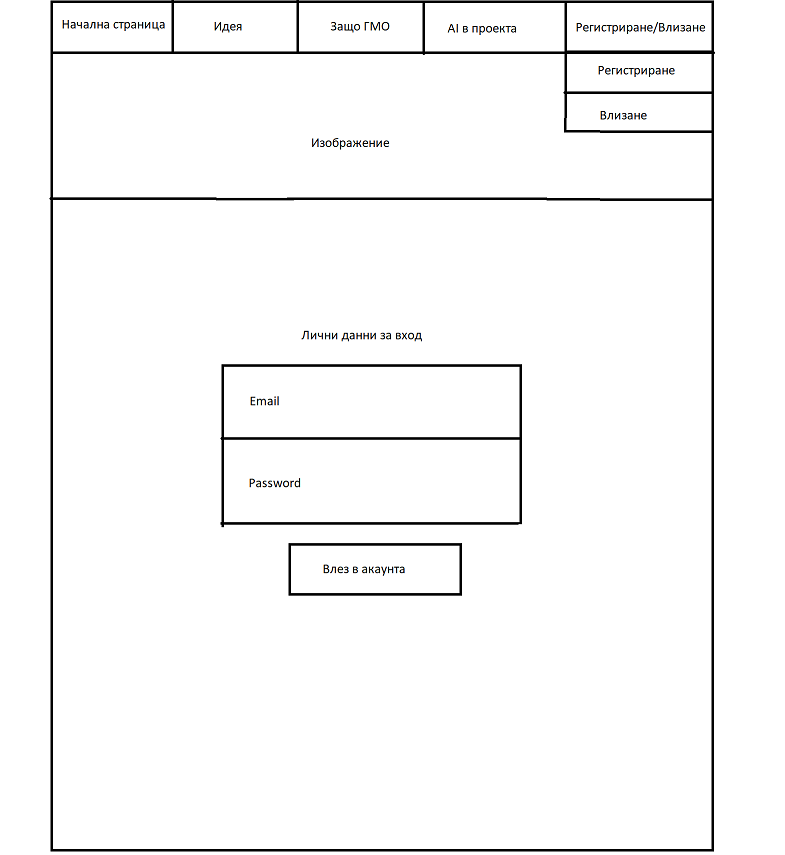
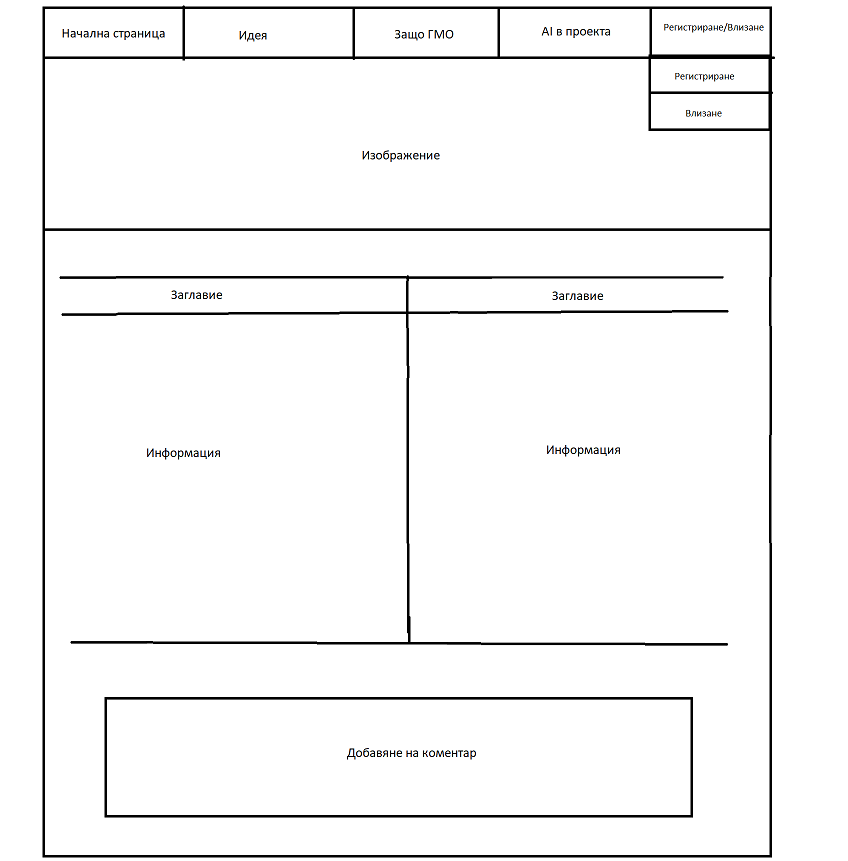
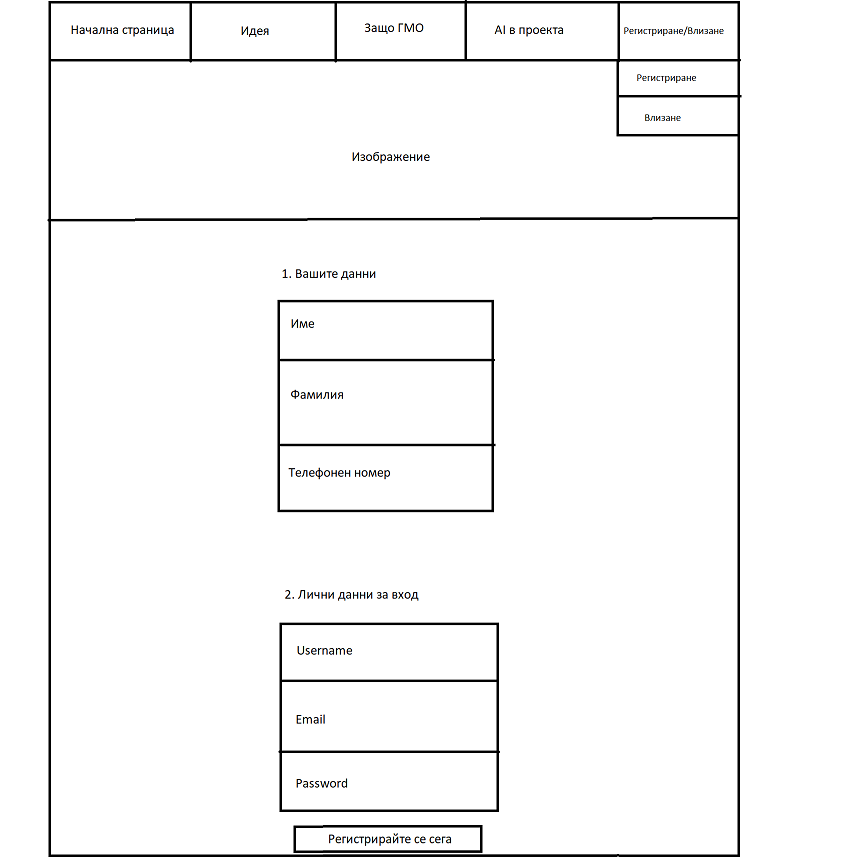
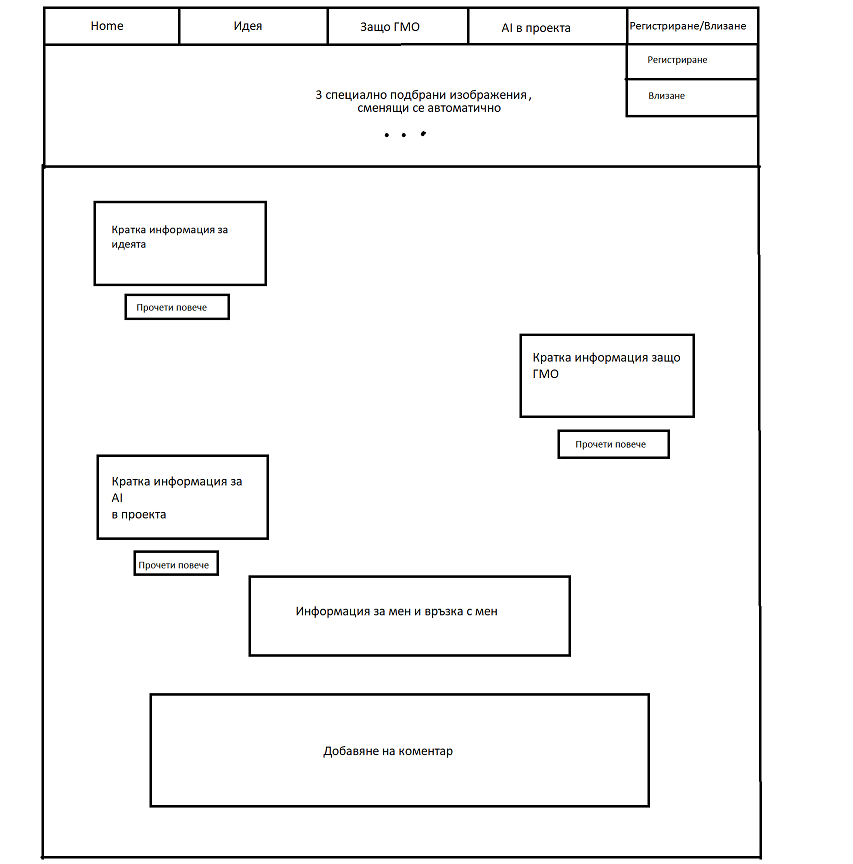
Приложената по-долу диаграма показва графично процеса.(*Фигура 3.1.2*)



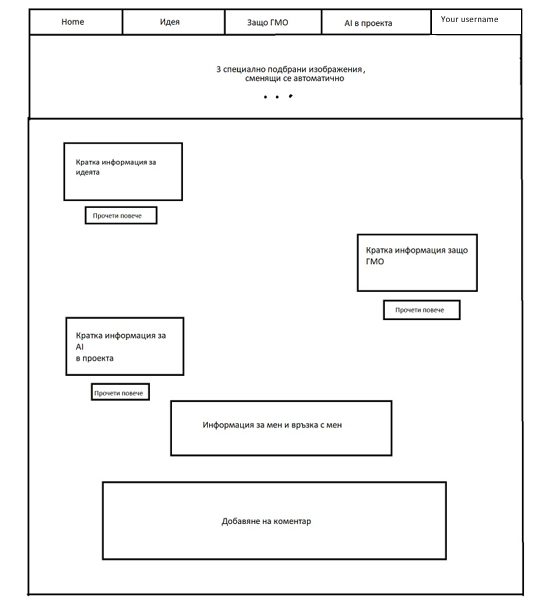
*Фигура 3.1.2*

## Примерен потребителски интерфейс

На схемите по-долу може да видите примерно решение за потребителския интерфейс(Изгледа е при нерегистриран или не влязъл в акаунта си потребител)(*Фигура 3.1*)

*Фигура 3.2*

На схемата по-долу може да видите примерно решение за потребителския интерфейс(Изгледа е при регистриран потребител)(*Фигура 3.2.2*)

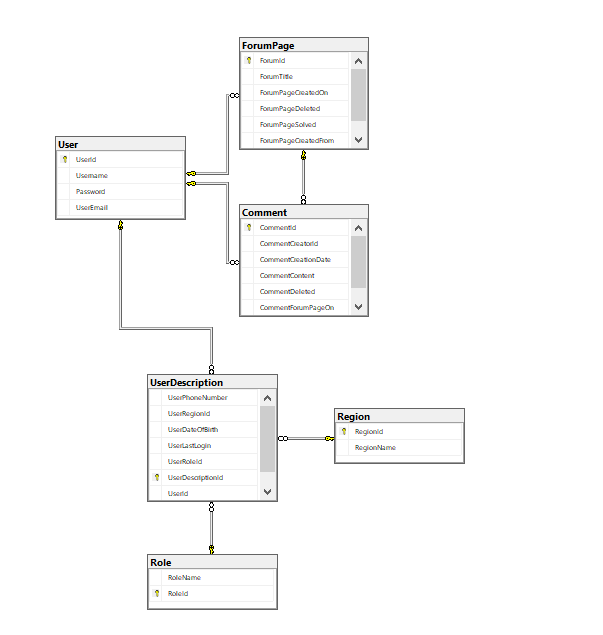


*Фигура 3.2.2*

## Диаграми на анализа

В базата от данни ще се съхранява информацията за регистрираните потребители, като ще се пази името, фамилията, имейла, телефона и паролата, които потребителя е въвел при регистриране.

Представената по-долу ER диаграма описва как ще работи и какво ще представлява базата от данни(*Фигура 3.3*)



*Фигура 3.3*

# Дизайн

За Web Server ще се използва Apache HTTP Server. Връзката на уебсайта и java заявките за базите данни ще се използва Аpache Tomcat.

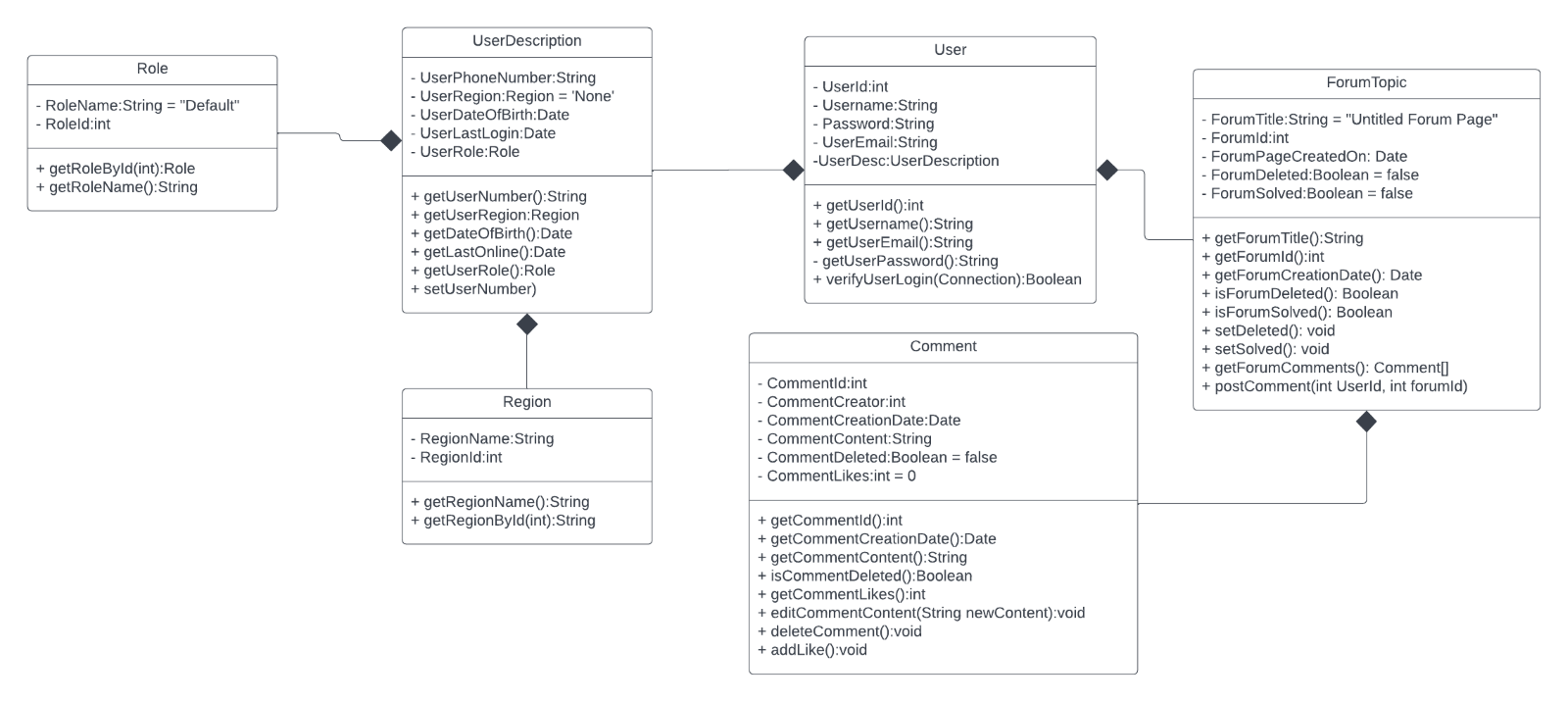
Езикът на който са писани заявките е Java, интегрираната среда за разработка на java заявките е “Intellij Idea (Community edition)” и “Eclipse”.

Проекта съдържа седем на брой класа, като основен клас от всичките е **User** класа.

Класът **UserDescription** наследява класа **User**.

Класовете **Role** и **Region** се използват в **UserDescription**, Comment се използва в класа **ForumTopic**.

Представената по-долу class диаграма показва класовете съдържащи се в проекта(*Фигура 4*)

*Фигура 4*

## Реализация на структура на приложението (3-layer), Разделение на кода според предназначението му

В софтуерното инженерство многослойната архитектура е архитектура от тип клиент-сървър, в която интерфейсът, обработката на приложения и съхранението и обработката на данни са логически разделени на отделни модули. Най-разпространената форма на многослойна архитектура е трислойната архитектура.

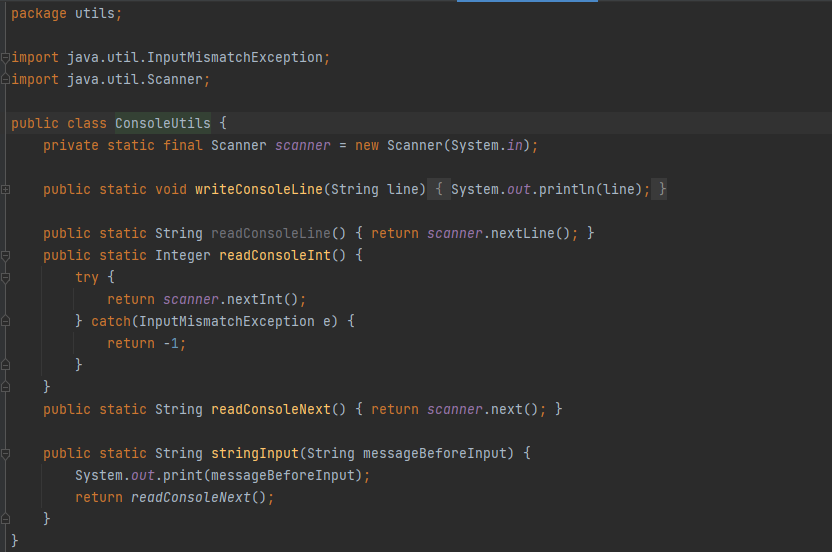
За проекта е използван трислоен модел, защото поради естеството му това би било най-оптималното решение.

Проекта се разделя на три основни слоя, като се добавят и допълнителни класове в него. В следващите точки ще разберете малко повече за класовете които са част от слоевете на програмата.

1. Първият слой на проекта съдържа части от кодове, които изпълняват дейността да показват самите данни (стойности) на потребителите. Клас който се счита от първия слой е:

* **ConsoleUtils** (Допълнителен клас съдържащ методи за отпечатването на данните в конзолата)

Снимката по-долу показва класа **ConsoleUtils**(*Фигура 4.1*)

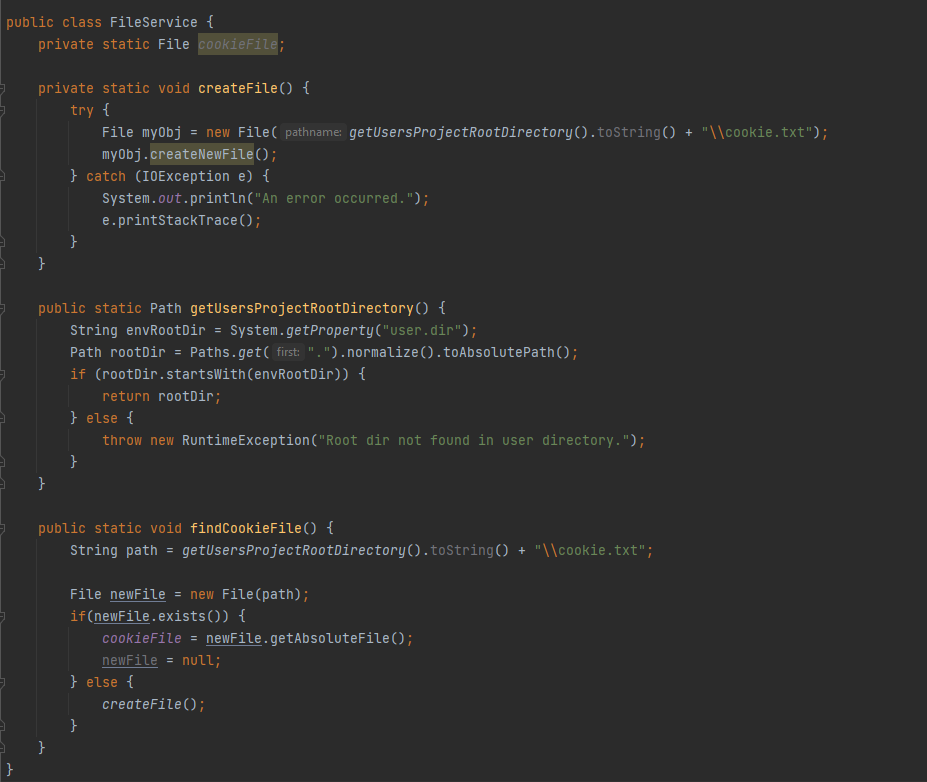


*Фигура 4.1*

1. Втория слой изпълнява дейността за обработката на данни чрез различни методи и препращането им на другите два слоя. Клас който се счита от втория слой е:

* **FileService** (Клас създаден с цел да запазва данните на потребител при влизането си в акаунта. По този начин ще избегне влизането в програмата(сайта) повторно.)

Снимката по-долу показва класа **FileService**(*Фигура 4.1.1*)

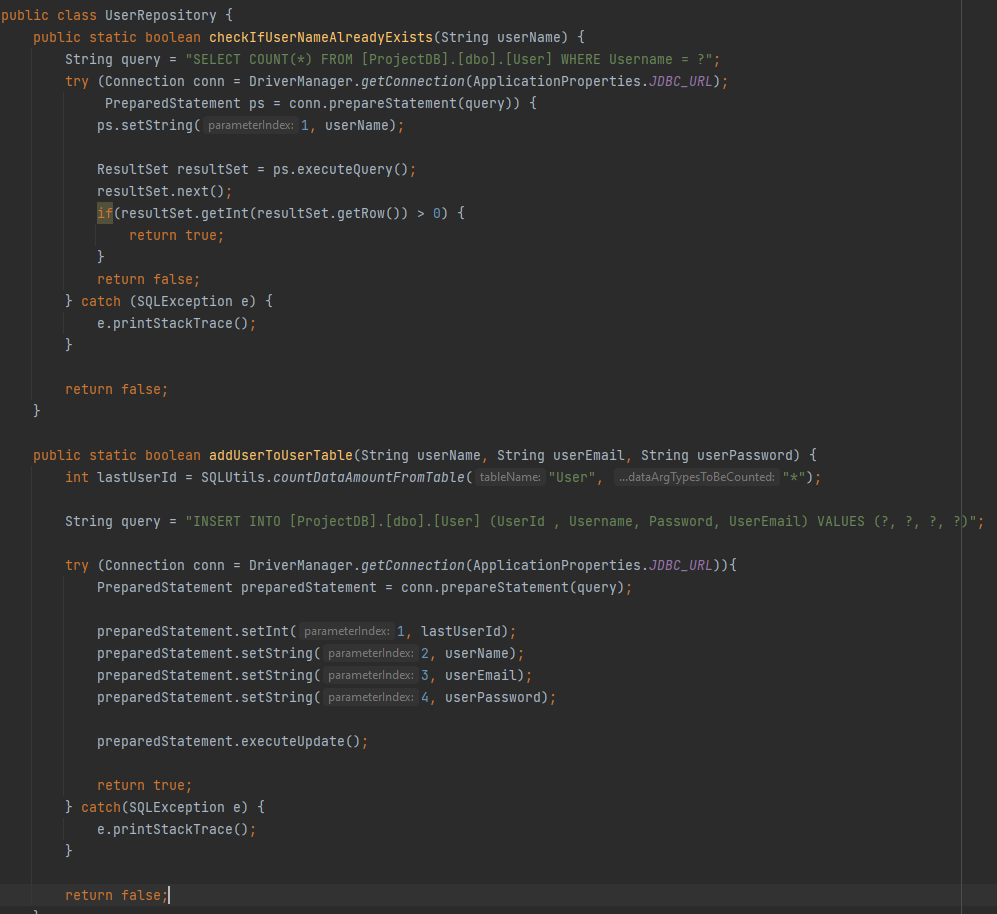


*Фигура 4.1.1*

1. Третият слой е свързан с това да изпраща и получава заявките от базите данни, за да може информацията в програмата да се актуализира. Клас който се счита от третия слой е:

* **UserRepository** (клас в който се създава, актуализира или премахва основният влязъл потребител в програмата)

Снимката по-долу показва класа **UserRepository**(*Фигура 4.1.2*)



*Фигура 4.1.2*

* На *Фигура 4.1.2* са показани и два метода “**checkIfUserNameAlreadyExists**” и “**addUserToUserTable**”.
* Първият клас се използва за това да провери преди регистрация дали съществува името което потребителя е въведено, като връща резултат от тип boolean.
* Вторият клас създава нов потребител, чрез три аргумента зададени в метода име, парола и имейл.
* Във всеки метод се съдържа поне една заявка, изпълняваща се след време, като базата данни връща резултат.

## Организация и код на заявките към база от данни

За организация на класовете и шаблоните са използвани наименования, които съдържат префикси или суфикси,както и могат да се разберат по местоположението им.Например:

1. Класовете свързани с изпращане на заявки до базите данни, завършват с суфикса “**Repository**”.
2. Класовете свързани с логическата (Бизнес) част на програмата, завършват с суфикса “Service”
3. Класа **ConsoleUtils**, който се използва за отпечатващата част на програмата, може да бъде намерен в папката utils (Utilities).

На снимките по-долу може да се видят заявките свързани с базите данни(*Фигура 4.2.1, Фигура 4.2.2, Фигура 4.2.3)*:



(Заявката сумира общия брой на съществуващи потребители с зададеното име)

*Фигура 4.2.1*



(Заявката добавя нов потребител в базите данни)

*Фигура 4.2.2*



(Заявката брои общия брой на съществуващите редове в таблица зададена от метода)

*Фигура 4.2.3*

## Наличие и интуитивност на потребителски интерфейс (конзолен, графичен, уеб)

В начален етап проекта е разработен чрез конзола, а в по нататъшен етап ще се разработи като уебсайт.

В уеб базираната система, потребителите ще могат да преглеждат свободно публикуваната информация.

Също така, могат да се регистрират или съответно да влизат в своите акаунти, ако вече имат такива.

Регистрираните потребители ще могат да пишат коментари под постовете, това всъщност е и тяхното предимство пред тези без акаунт.

# Тестване

Тук се *включват тестовите случаи* и какви видове тестване предвиждате в реалното изпълнение на проекта, напр. с колко и какви документи, в какви браузъри, с какви приставки, и т.н.

# Заключение и възможно бъдещо развитие

В заключение, обобщете резултатите от работата ви по проекта, както и предимствата и ограничеността на използваните технологии / езици / методи. Укажете какви алтернативи могат да се използват и техните предимства и недостатъци. Опишете каква е използваемостта на подобни решения в практиката и какво бихте предложили като насоки за бъдещо развитие на вашето решение.

# Използвани литературни източници и Уеб сайтове

Използвайте вградената функционалност на Word: References > Citations & Bibliography

1. Уеб сайт на ….., адрес ….
2. Уеб сайт на ….., адрес ….
3. Уеб сайт на ….., адрес ….
4. Уеб сайт на ….., адрес ….
5. Литературен източник 2
6. Литературен източник 3
7. Литературен източник 4
8. Литературен източник 5

# Приложения

При необходимост можете да добавите и допълнителни секции под формата на апендикси. Таблица с диаграми, таблици и графики

*Забележка:*

1. *Документацията на проекта се предава само в електронен вид в MS Word, чрез качването на архив с документа и останалите файлове по проекта, в задание за предаване на проект, в канала на екипа в Teams.*
2. *Кода на проекта, базата данни и документацията трябва да са налични в репозитори в GitHub, което е копие на заданието генерирано в организацията.*

# Критерии и показатели за оценяване

|  |  |  |
| --- | --- | --- |
| **Показател** | **точки** | **срок** |
| 2. [Цели и обхват на софтуерното приложение](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Цели_и_обхват)  3.1 [Потребителски изисквания и работен процес](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Потребителски_изисквания_и) | 5  5 | 04.03.2022 |
| 3.2 [Примерен потребителски интерфейс](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Примерен_потребителски_интерфейс)  3.3 [Диаграми на анализа](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Диаграми_на_анализа)  3.4 [Модел на съдържанието/данните](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Модел_на_съдържанието) | 5  5  5 | 18.03.2022 |
| 4.1 [Реализация на структура на приложението (3-layer),  Разделение на кода според предназначението му.  Допълване на Class диаграми/3.3/](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Дизайн). | 10  10 | 16.04.2022 |
| 4.2 [Организация и код на заявките към база от данни](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Дизайн). | 15 | 30.04.2022 |
| 4.3 [Наличие и интуитивност на потребителски интерфейс (конзолен, графичен, уеб)](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Дизайн). | 10 | 31.05.2022 |
| 5. [Наличие и организация на автоматизирани тестове](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Тестване). | 15 | 15.06.2022 |
| 6. Организация на проекта в система за контрол на изходния код и употреба на добри практики (merge requests, code reviews, branching strategy) | 10 | 25.06.2022 |
| 1. [Въведение. Ниво на завършеност на проекта](file:///O:/Admin204.1/Проекти/2021-2022/Ученически%20практики%202/Ученическа%20документация/user_name_Software%20Development.docx#_Въведение) |  | 30.06.2022 |
| Документация на проекта (XML comments, wiki, etc.) | 5 | текущо |
| Презентация на проекта |  | 30.06.2022 |
| Общо | 100 | Финал на първа фаза |