**ПРОФЕСИОНАЛНА ГИМНАЗИЯ ПО МЕХАНОТЕХНИКА, ЕЛЕКТРОНИКА, ТЕЛЕКОМУНИКАЦИИ И ТРАНСПОРТ**

**„ХРИСТО БОТЕВ“, ГР. ШУМЕН**

**ДИПЛОМЕН ПРОЕКТ**

**Тема: Image Gallery „GlimpseHub”**

**Разработил: Марин Неделчев Неделчев**

**от професия код 481030 „Приложен програмист”**

**специалност код 4810301 „Приложно програмиране”**

**Ръководител-консултант: инж. Николай Христов**

2024 г.

С Ъ Д Ъ Р Ж А Н И Е

I. Предназначение на приложението и проблеми, които то решава

1.Предназначение

2. Проблеми, които приложението решава

II. Функционалности и правата на достъп които приложението предлага

1. Нива на достъп и правомощия в приложението

III. Използвани технологии при разработката

1. Обща информация за приложението

2. Програмни продукти използвани при разработката на приложението

3. Програмни езици използвани при разработката на приложението

4. Библиотеки използвaни при разработка на приложението и тяхното предназначение.

IV. Структура на приложението и организация на кода

1. Структура на приложението

2. Разделеняне на кода, функции и предназначение на всеки раздел

3. Достъп и зависимости на отделните проекти в Solution, видимост

4. Съображения при избора на връзки между класовете

V. База данни

1. Избор на база данни

2. Подход при разработката и внедряване на базата

3. Създаване на база данни и реализация в проекта

4. Модели-описание на релации и организация на данни те във всяка таблица (модел)

5. Особености и разлики в моделите и таблиците от базата

6. Типове релаци и между таблиците им оделите

VI. Фронт-енд визуализация и контролери

1. Razor Pages

2. Razor Views

3. Статични ресурси.

VII. Функционалност на приложението която, следва да се имплементира, надграждане. 1. Галерии

2. Снимки

3. Събития

4. Секция отзиви от клиенти

5. Коментари под даден филм от потребители, които са били на прожекция на филма

6. Връзка потребител - администратор

**I. Предназначение на приложението и проблеми, които то**

**решава.**

1. Предназначение:

Уеб приложението „GlimpseHub“ представлява платформа за споделяне на изображения, която цели да създаде виртуална общност от любители на фотографията и визуалното изкуство. Създадено в контекста на дипломен проект, то предлага възможност за публикуване на снимки и галерии от потребителите, както и за обмен на рейтинги и коментари.

2. Проблеми, които приложението решава:

Приложението ми, “GlimpseHub” предлага опцията да се споделят снимки с хора по целият свят и да се пазат в галерии независимо дали скрити или публични.

**II. Функционалности и правата на достъп които**

**приложението предлага:**

1. Нива на достъп и правомощия в приложението:

1.1. Гост GUEST – това са всички потребители, които достъпва приложението без да имат регистрация в него. Тяхната анонимност е

гарантирана. Има права да разглежда наличните филми както и

детайлите в тях. Може да вижда секцията контакти.

1.2. Потребител USER – представлява регистриран потребител,

притежава всички права на гост потребител като има възможност да

запази лични данни (имейл и парола ) като и да ги променя в

последствие.

4. Библиотеки използвани при разработка на приложението и тяхното предназначение.

4.1. Библиотеки при сървърната част (back-end):

* Entity Framework – Представлява ORM предоставящ възможност за работа с база чрез MS SQL Provider, като по този начин лесно може приложението да се пригоди към друга база, например Postgre или МySQL. Кодът не е обвързан с конкретният SQL език пряко.
* Мicrosoft Identity – Представлява набор от класове, които се материализират в базата и дават базова функционалност за управление на потребители, като когато това не е достатъчно могат да бъдат разшир явани както в случая е разширен класът AppUser.
* LINQ – Набор от Extension методи върху колекци и позволяващ работа с тях. Използва се в съвкупност с Entity Framework като работата с колекции класове се превежда от ORM до заявки към базата на съответният език избран чрез Provider.

4.2. Библиотеки при визуалната част (front-end):

Тези библиотеки се зареждат през CDN връзки (линкове) с цел по - бързото им сваляне от клиент а предвид разположението им на множество сървъри.

* Bootstrap – Съдържа набор от CSS класов е дефиниращи правил а за визуализация на html. Цели да позволи писане на HTML без да се налага писане на CSS а вместо това се присвояват имената на класовете от библиотеката наразлични HTML елементи.
* Font Awesome – Съдържа набор от CSS класове дефиниращи как да изглеждат иконки от тип < i class =” …” >. Като това позволява да се визуализират иконки в приложението.

**IV. Структура на приложението и организация на кода .**

1. Структура на приложението:

Приложението е разработено на база основа ASP.NET MVC като е разширено и преструктуриране за да позволява скалиране и менажиране на кода и отстраняване и локализиране на бъгове в процеса на разработка и експлоатация.

2. Разделеняне на кода, функции и предназначение на всеки раздел:

2.1. Application представлява Console Application. Ролята му е да стартира приложението и да управлява и използва всички останали компоненти в целият Solution. В него се съдържат следните компоненти:

* wwwroot: В него се съдържат статични материали, като изображения, javascript и CSS файлове, които се достъпват от Views.
* Areas: Съдържа скафолднати Identity Razor Pages, които са разширени допълнително за да прилягат на целите на приложението.
* Controllers: Съдържа класовете контролери които имат за цел да управляват кои Views да бъдат върнати на потребителя както и да ги снабдяват с нужните за визуализирането им данни.
* Views: Съдържат организирани в папки за да бъдат откривани по конвенция \*.cshtml файлове (Razor views), които се достъпват и връщат през методите в контролерите.
* appsettings.json – (традиционно тук се съдържа ConnectionString към съответната база както и друга чувствителна информация).
* Program.cs – Представлява стартируемият клас от който се стартира приложението в него се съдържа „ public static void Main (string[] args)“метода.

2.3. Data.Models – съдържа всички Entities, които имат репрезентация в базата с данни. Разделението на отделните класове-модели е в подпапки, но с цел опростено намиране всички са регистрирани в общ namespace Data.Models. Библиотеката съдържа следните раздели:

* Enums – Съдържа енумерации, които се ползват в класовете, на практика в тях се съдържат дефиниция (шифър), как да се интерпретират числовите стойности на отделните колони в базата със стойности от тип енумерация.
* Models – Съдържа базовите абстрактни класове наследени от моделите, както и модели на класове репрезентиращи таблици в базата данни.

2.4. Data – Папка съдържаща информацията нужна на ORM Entity Framework как да направи и свърже Models със таблиците от базата данни:

 Data – Съдържа ApplicationDbContext, наследник на DbContext. Указва кои таблици ще бъдат създадени на базата на съответните класове от Models, също така указва в метода „protected override void OnModelCreating(ModelBuilder builder)“ чрез fluent API правилата които не могат да се дефинират чрез конвенционално именуване или атрибути. Например композитни ключове за mapping таблици.

* ApplicationDbContext.cs – Клас съдържащ метод който да връща конфигуриран ApplicationDbContext (в него трябва да е зададен ConnectionString за да се осъществи връзка с базата данни.
* Migrations – Директория, в която се съдържат ми грациите, които отразяват промените в структурата на базата при началното и създаване както и в процеса на промяна по времена експлоатация на приложението.

4. Съображения при избора на връзки между класовете:

При по строяването на връзките “dependencies”се съблюдават две

правила.

 Да може да функционира приложението тоест във всяка

библиотека-проект да има достъп до ресурси(класове)

намиращи се в други библиотеки-проекти.

 Да се избегне CircularReference. Не се допуска две библиотеки

да са видими взаимно пряко или чрез посредник, вместо това

трябва достъпът да бъде иерархичен. По тази причина видно от

зависимостите може да се заключи, че на върха на йерархията на

проектит е в Sоlution се намира Application конзолното

приложение от което се стартира цялото приложение.

**V. База данни.**

1. Избор на база данни:

При разработката на приложението е избрана релационна база данни

Microsoft SQL Server и приложение за управлението и Microsoft SQL

Management Studio, в което може да се визуализира базата както и да се

пишат Transact SQL заявки към базата.

2. Подход при разработката и внедряване на базата:

2.1. Подходи при създаване на нова база данни:

* Data Base First – При този подход базата се изгражда посредством скрипт написан и изпълнен на Transact SQL, като всички правила, Ограничения, процедурии т.н. се дефинират в SQL. След като базата е готова на всяка таблица от нея трябва дa се създаде отразяващя клас, който да репрезентира данните от базата. Създаването на тези класове е възможно да се изпълни и чрез Scaffold автоматично.

Основно предимство на този подход е, че базата може да се дефинира по -оптимално и ясно в T-SQ L. Могат да се използват всички “features” на

конкретният SQL език вместо само такива валидни за всички SQL езици.

Недостатък на този подход е че базата дефиниран а през T-SQL е обвързана с конкретният избор на SQL сървър, изисква се познание в избраният SQL език.

 Code First – При този подход б аз а т а с е из гр а жд а с л ед и въз

о сн о в а н а кл а с ове в C #. З а да се с ъзд ад е б аза т а, с е г ен ери ра т

м иг р ации к оито се п рил аг а т къ м ба з ат а. Всяк а т ак ав а миг р ация

с ъд ър ж а а вто м атич но г ен е рир ани T-S Q L пр а вил а . К а то мо г ат д а с е

из вър ш в ат пр о мени по к л а со в ет е и д а с е д опъл в ат м иг р ациит е в

п ос л ед ст ви е.

О с но вно пр е ди мс тв о н а т ози подх о д е , ч е O RM ко р е сп ондира

с избр ани ят S Q L с ър въ р, ко е то по зв ол яв а н а п ро гр а ми ст а д а н е

изп олз в а чи ст S QL и п ре до ст а в я въз мо жно с т с мно го м а лко ус и лия

д а с е см ени S QL с ъ р в ъ ра н ап рим е р с Po st gr e или M yS Q L в

п ос л ед ст ви е .

Н е д ос та тъцит е на т ози п одх о д с а:

П он е ж е с е раз чита н а Pr ov id e r к л ас к ойт о д а г ен ери ра S Q L

к од, с е изи ск в а п ознани е к ак д а с е ук а ж а т пр а вила при с а мит е

кл а с о ве кои то д а д о ве д ат до пр а вилн а инт е рпр ет аци я н а к л а со в ет е.

П ри м ер з а то в а с а а три б ут и , Flu e ntA p i п р ави л а, или и м ено в ан е н а

с в ой ст в ат а н а кл а со в ет е по кон в енция .

Д р уг н ед о ст атък е , ч е ако с е изи ск в ат ф ун кцион а лно с ти от

б аз ат а к ак вито с ам о кон кр етн ат а ба з а п р ед ла г а, т о то в а н е е на й -

уд а ч н ият в ари ант з ащ ото P r ovi d e r кл а с ов ет е з а все ки тип б аз а

п одд ър жа т об щи инт е рф ей си, то е ст са м о о бщ а ф ун кци он алн о ст.

Дипломна работа стр.13

2 . 2. Из бр ан подх од и об о сно вк а:

З а р аз р аб отк ат а на п рил ож ени ет о е п ре дпо ч ет ен подх о д а C od e

Fi r st п он еж е в п ро це с а н а р азр а ботк а щ е с е пр о м енят кл а со в ет е и

щ е с е раз чи та н а ми гр ации з а д а с е отр аз ят т ези пр о м ени, същ о

т ак а C od e Fi r st пре д л аг а въз мо жно ст д а с е р аз р аб оти п рило ж ени ет о

б ез изпо лзв ан е н а T-S Q L а фок ус ъ т с е изм е ст в а в п ос ок а н а C#

кл а с о ве те , къ де то м о ж е д а с е възп олз в а п ро г ра ми ст ът о т In t el is en s e

фун кциит е на Vis ua l St udi o 20 22 .

З а н екон в енциона лни те пр а вил а с е ра зши ря в а м ет одът

p r ot e ct e d ov e r rid e v oi d O nMo d elC r e ati n g( Mo de l Bu ild e r bu il de r )

Т ук с е з ад а в ат п р ав ил а , к ат о к о мпозитни клю ч о ве н апр им е р .

В д опъ лн ени е към кон в енци она лнит е п р ави л а и Flu e nt Ap i

п ра ви ла т а, н а ме с т а с а изпо лзв ани ат риб ут ни о бо зна ч ени я в

кл а с о ве те д е финир а щи т аблици те мо де ли \* (M od e ls ).