**Курсов проект**

**по**

**Управление на софтуерни проекти**

**Изработили: Стефан Димов,**

**Димитър Иванов,**

**Божидар Маргенов**

**Факултетен №: 61662201,**

**61662202,**

**61662203**

**Специалност: СИТ**

**Курс: III**

**Група: 4а**

**Начален екран**

Съдържа списък с добавените песни.До него не намират бутон за добавяне и за първоначалното зареждане на данните от файла .За всяка песен излиза информация с име,изпълнител,година,жанр и дължина.В долен ляв ъгъл се намира меню,чрез което можем на сортираме.Долу в ляво се намира списък с песните в азбучен ред или по дължината им.

**Меню за добавяне на песни**

В него се въвежда име,изпълнител,година,жанр и дължина.При натискането на бутона песента се добавя в списъка.

**3.Обхват на разработка**

Продуктът трябва да помага на потребителя да търси песни различни критерии.

**4.Изисквания към програмния продукт**

**4.1 Функционални изисквания**

**4.1.1. Модул за въвеждане на информация за песен**

* Възможност за въвеждане на име
* Възможност за въвеждане на изпълнител
* Възможност за въвеждане на година
* Възможност за въвеждане на жанр
* Възможност за въвеждане на дължина

**4.1.2. Модул за търсене на песен**

* Възможност за сортиране по име
* Възможност за сортиране по изпълнител
* Възможност за сортиране по година
* Възможност за сортиране по жанр
* Възможност за сортиране по дължина

**5. Системна документация**

Изпълнителят трябва да подготви следната документация:

• „Описание на структурата и принципа на работа на системата”

• „Ръководство за конфигуриране, контрол и поддръжка на системата”

Всички документи е необходимо да бъдат на български език.

Всички документи трябва да бъдат предоставени в два екземпляра, на хартиен и електронен носител.

# 6. Софтуерна поддръжка –възможност за разработване на допълнителни модули

6.1.Възможност за разработване на допълнителни модули:

6.1.1. Със закупуване на софтуерния продукт възложителят получава право над програмния код, който при желание от негова страна, може да бъде доразработен от изпълнителя, включвайки нови функционалности и допълнителни модули или промяна на съществуващи такива – създаване на нови версии на програмния продукт.

6.2.Поддръжка

* + 1. Според подписания с възложителя договор изпълнителят поема отговорност за безплатна поддръжка в рамките на първия месец. След изтичане на този период поръчителят трябва да заплаща сума от 350 лева месечно за поддръжка (10 % от цялата цена на продукта).

**7.Система за контрол на версиите**

Система за контрол на версиите се ползва за записване на промените по даден файл или колекция от файлове. Тя позволява да се запази история на промените, да се върне предишна версия, да се добави описание на промените (като например защо и на какво е извършена промяната и други).

**Git**

За разработването на този софтуерен продукт е използвана децентрализираната система за контрол на версиите – Git. Git е впечатляващо бърза, ефективна и разполага с невероятна branching система за нелинейна разработка, като това е само част от причината тази система да бъде избрана за настоящия проект заедно с останалите, които ще бъдат споменати по-нататък . С Git много потребители могат да работят едновременно по един и същ файл, без това да доведе до бъркотия и хаос в съдържанието на файла. Git е създаден и разработен през април 2005г. от не кой да е, а Линус Торвалдс(човекът зад Linux ядрото). Торвалдс създава Git, за да го ползва за разработката на неговия проект Linux ядро. От тогава до днес Git се развива и подобрява от множество потребители. Предимствата, които предоставя тази система са свързани най-вече със следните характеристики :

* Скорост
* Опростен дизайн
* Мощна поддръжка за нелинейна разработка (хиляди паралелни клонове код)
* Напълно разпределена работа
* Възможност да облужва ефективно големи проекти като Linux ядрото (по отношение на скорост и обем на данните)

Друго нещо, което трябва да се спомене е, че Git съхранява и разглежда информацията съвсем различно в сравнение с другите системи, макар потребителският ѝ интерфейс да е подобен с техните. Ето и част от разликите :

**Snapshot-и вместо разлики**

Основната разлика между Git и другите системи за контрол на версиите е начинът, по който Git третира данните. Концептуално, повечето други системи записват информацията като списък от файлово-базирани промени, тоест тези системи виждат информацията, която съхраняват като колекция от файлове и промените направени във файловете във времето. Вместо това, Git възприема своите данни по-скоро като колекция от snapshots (моментни снимки на статуса) на една миниатюрна файлова система. Всеки път когато къмитвате или записвате статуса на вашия проект в Git, системата най-общо казано прави снимка на това как изглеждат файловете ви в този момент и съхранява референция към този snapshot. Това е важна разлика между Git и почти всички други VCS.

**Почти всяка операция е локална**

Повечето операции в Git се нуждаят само от локални файлове и ресурси - общо взето не се нуждаете от информация намираща се в мрежата. Например, за да ви покаже историята на проекта, Git не се нуждае да контактува със сървъра а просто чете директно от локалната си база данни и вие виждате историята почти незабавно. Ако желаете да видите промените в даден файл между текущата му версия и тази отпреди месец, Git ще направи локална калкулация на разликите, вместо да трябва да пита отдалечения сървър за това или да трябва да издърпва по-стара версия от сървъра и едва след това да калкулира разликите локално. Това също значи, че можете да правите почти всичко когато сте офлайн или когато VPN връзката ви не работи например.

**Git само добавя данни**

Когато се прави нещо в Git, тези действия само добавят информация към базата данни на Git. Трудно е системата да направи каквото и да е без то да може да бъде възстановено или пък да се изтрият данни безвъзвратно. Разбира се, подобно на всяка друга VCS, могат да бъдат загубени или объркани промените, които не са били къмитнати, но веднъж направили snapshot-а в Git - е много трудно да се загубят данни, особено пък ако редовно се изпраща базата към отдалечено хранилище.

**Мониторинг на интегритета на данните**

Преди да запазите файл,индексът се определя под формата на контролна сума, изчислена директно от Git. Какво представлява контролната сума? Тази стойност се изчислява, като се използват специални алгоритми и се използва за проверка на целостта на данните по време на съхранение и предаване. Невъзможно е да се промени нищо без знанието на Гит и това е важна част от философията на системата.

**Предимството на Git пред SVN/CVS и TFS**

Използването на Git решава най-големия недостатък на другите две – в случай на повреда в централния или локалния компютър, всички данни на проекта могат да бъдат загубени, тъй като се намират на едно физическо място. Предимството на Git е, че тази система за контрол на версиите е децентрализирана поради, което проектът се сваля и копира локално на компютъра на всеки участващ потребител, като така се създава бекъп. В случай на необходимост, проектът може да се възстанови от локалното копие на някой от потребителите.

**Недостатъци**

* Състои се от много команди и опции и е нужно високо ниво на разбиране, тъй като някои команди и аргументи са непоследователни до някаква степен.
* Не достатъчно добра поддръжка под Windows
* Слаби инструменти за бинарни файлове
* Не поддържа празни директории
* Липса на ограничение за контрол на достъпа

**8.Система за управление на проекта**

**Trello** е безплатно уеб-базирано приложение за управление на проекти, подпомагащ увеличаването на производителността при изпълнение на колективна работа, с понастоящем 4 милионаабонати. През 2013г. стартира платената услуга за бизнес класа. Трело може да се използва не само за управление на проекти, но и за организиране на личните ежедневни задачи.

Trello използва методологията Канбан, която представлява система за управление на проекти, разработена от бившия вицепрезидент на Тойота, Таиичи Она. Trello предоставя дъски (boards), списъци (lists) и карти (cards), за да се организират и приоритизират проектите. Най-общо една дъска представлява един проект. Към нея може да се изградят няколко на брой списъка – например TODO, IN PROCESS, DONE. В тези списъци се поставят задачи – карти. След създаването на задача, към нея могат да се добавят коментари, да се прикачат файлове, да се изготвят подсписъци за проверка (checklists), да се задват крайни срокове и др. Картите имат предна и задна част. В списъка на таблото се вижда само предната им част – техните заглавия и основна информация, която насочва какво съдържа задачата. Тук можете да се добави и снимка, както и да зададете различни цветове на отделни задачи – за да се групират по определен признак, или просто да се различават по-лесно. Задната част се вижда, когато се влезе в конкретната задача. Там вече има изключително много възможности, сред които чеклистове, задаване на крайни срокове, следене на активността, добавяне на коментари, закачане на файлове, архив и т.н.

**Предимства**:

* Изключително интуитивен софтеур за управление на проекти.
* Бърз поглед на подреждането на картите позволява на потребителите да знаят докъде е стигнал проекта и какви са задачите им за следващия.
* На предната част на картата може да се напише основна. На “гърба” обаче може да се попълнят различни данни, отнасящи се до конкретната задача, като всичко това може да бъде организирано лесно и визуално.

**Недостатъци**:

* Простотата и минимализмът не позволяват проследяване на проектите в по-дълбока детайлност.
* Не е подходяща система за управление на сложни проекти.
* Няма възможност да се видят задачите, подредени по потребител или краен срок.

**9.Методология на разработване**

**Екстремно програмиране, Extreme Programming, XP**

Методологията на разработване, която сме избрали е Екстремно програмиране, тази методология спада към Гъвкавите технологии, които притежават следните пет принципа:

* Реализация на „разрастващо се" (инкрементално) разработване, като изискванията за всяка следваща итерация се преформулират въз основа на оценяване на текущата;
* Регламентиране на стил на работа, основан на компетентност, мотивираност и екипност; стимулиращ творчеството психологически климат;
* Приемане на неизбежността на промените, съпътстващи софтуерните проекти и адаптиране на техниките на разработване към тях;
* Придържане към опростени конструкции и схеми на работа, с цел минимизиране на ресурсите за осъществяването им.
* Активно включване на потребителите в процеса на разработване на Софтуера;

**Други характеристики на този вид технологии са :**

• Гъвкавите технологии не предлагат универсално решение на проблемите в софтуерното производство.

• Те са подходящи, ако в организацията е налице нужният стил на работа, при това за разработка на не „критичен” софтуер.

Причината за избора на тази методология е, че тя е предназначена за малки и средни колективи, разработващи софтуер при неясни или бързо променящи се изисквания, както и заради скоростното получаване на конкретен резултат. Тъй като проектът може да претърпи промени в бъдеще, тази методология е предпочитана пред стандартните методологии, при които цената за реализация на промените е скъпоструваща.

Екстремното програмиране в частност притежава следните характеристики:

* кратки цикли
* метод на инкрементално планиране(стъпка по стъпка)
* непрекъснато тестване като тестовете се задават от потребителя
* непрекъсната интензивна комуникация
* тясно сътрудничество м/у програмистите
* ще могат да променят насоката на проекта по време на реализацията му без това да бъде свързано с големи разходи.

**Екип роли и задачи**

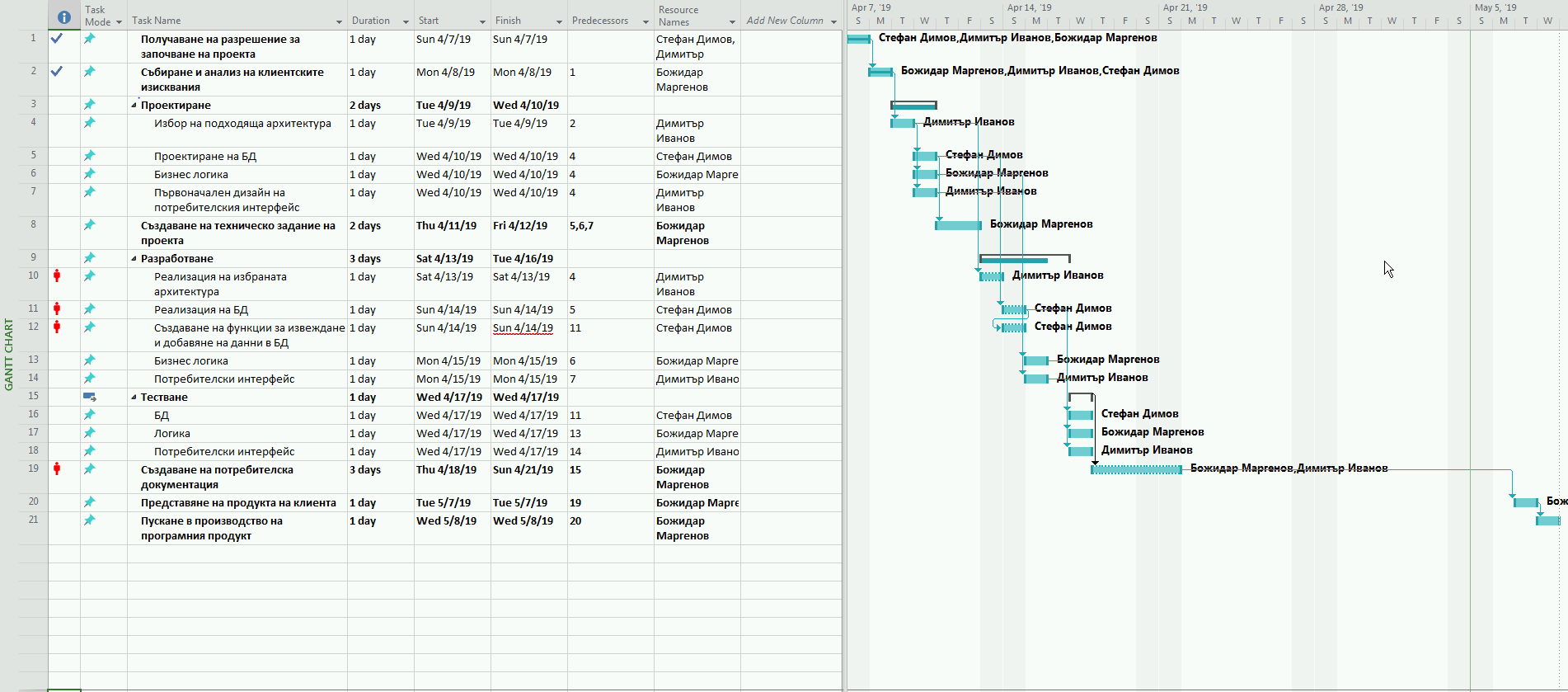
**1.Екип:**

1. Стефан Димов - СД
2. Божидар Маргенов - БМ
3. Димитър Иванов - ДИ

**2.Роли:**

* Разработчик на бизнес логика: БМ
* Разработчик база данни: СД
* Разработчик потребителски интерфейс: ДИ
* Тестери : БМ, СД, ДИ

**3.Задачи:**

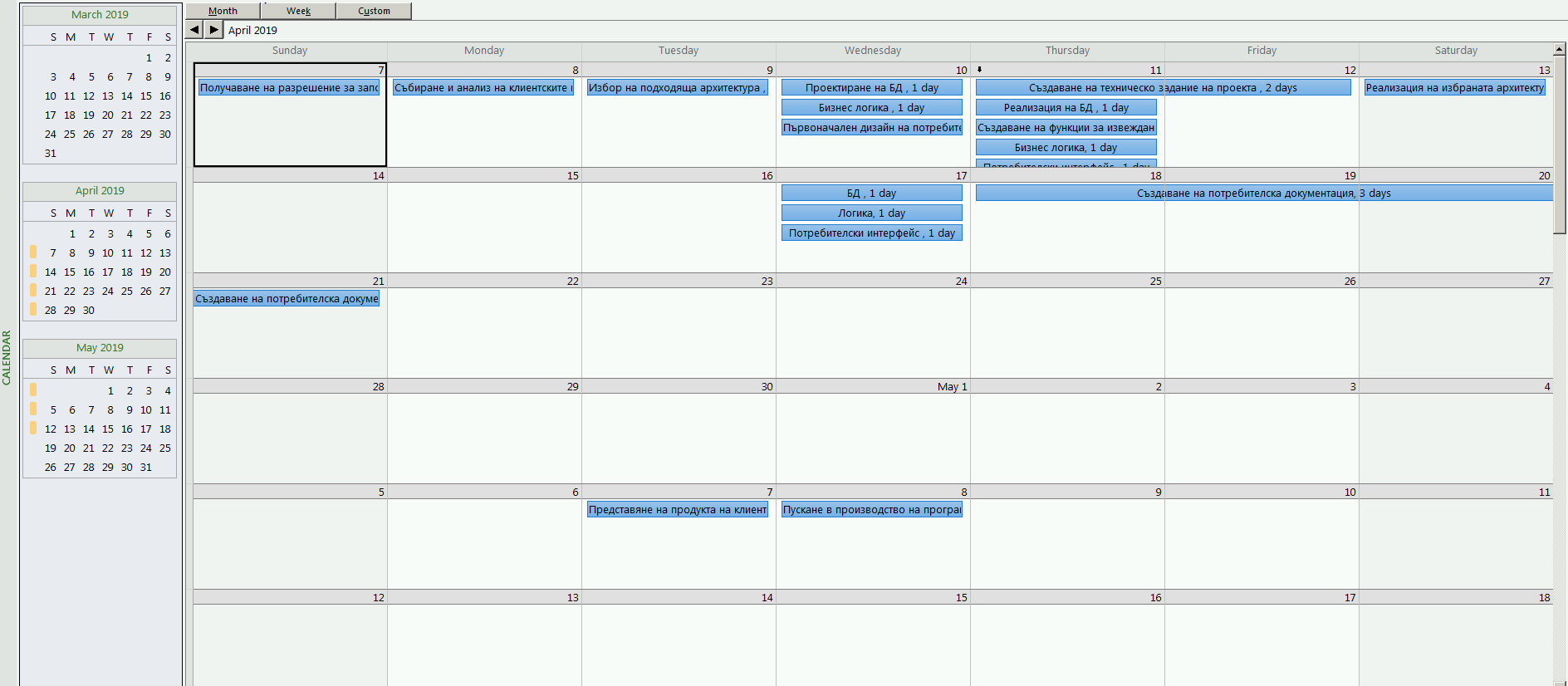


**Диаграма на свършената и оставащата работа**

**Диаграма за времето за изпълнение на задачите**

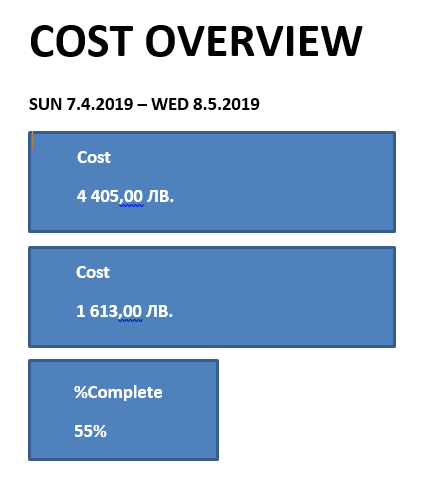
**Диаграма за времето за изпълнение на задачите**

**График за изпълнение на проекта**

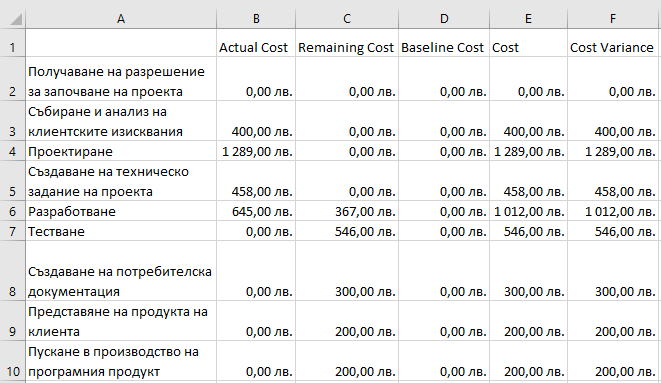


**Бюджет на проекта**

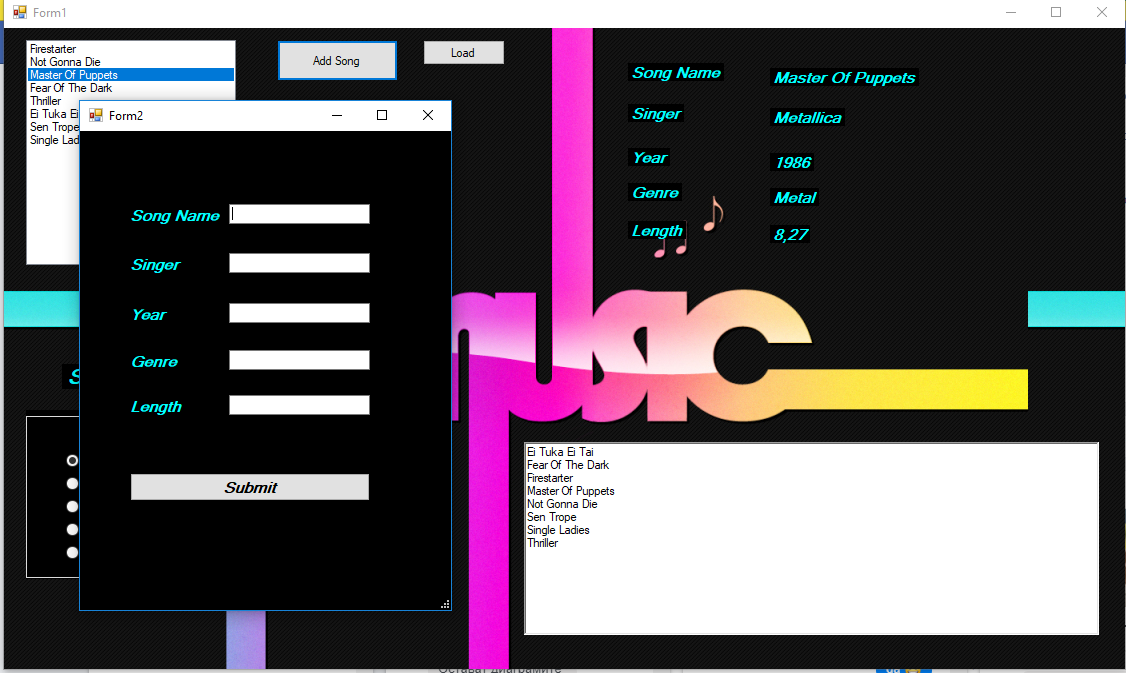
**Бюджет на проекта**



**Бюджет на проекта**



**Краен продукт и интерфейс на приложението**



**Възможност за бъдещата разработка**

* Custom анимации
* Създаване на вграден player
* Създаване на сървърна част
* Превръщане в мултипотребителско приложение