**Курсов проект**

**по**

**управление на софтуерни проекти**

**Изготвили:** Добромир Начев, Александър Стоименов

**Факултетен номер:**18621700,18621697

**Специалност:** СИТ

**Курс:** III

**Група:** 2б

Съдържание

[1.Бюджет 3](#_Toc73819327)

[2.Предмет на техническото задание 3](#_Toc73819328)

[3.Обхват на разработката 7](#_Toc73819329)

[4.Система за контрол на версиите 7](#_Toc73819330)

[5.Система за управление на проекта 9](#_Toc73819331)

[6.Методология на разработване 11](#_Toc73819332)

[7.Екип, роли и задачи 13](#_Toc73819333)

[8. Microsoft Project 17](#_Toc73819334)

[9.Бизнес логика 21](#_Toc73819335)

[10.Изводи 22](#_Toc73819336)

# 1.Бюджет

Бюджетът на проекта е изчислен въз основа на разходите, които трябва да се покрият по време на разработването на проекта. В изчисленията са включени почасовото заплащане на разработчиците въз основа на задачите, които те трябва да изпълнят – анализ на изисквания, планиране и реализация на плана на проекта. Към изчисляването на бюджета също така спадат и използваните от разработчиците ресурси, статична сума за поправка на дребни компютърни проблеми, 10% за справяне с настъпил риск и 10% за бъдеща поддръжка и добавки към приложението. Взимайки предвид всичко това, общата стойност на проекта възлиза на **3062.40 лв.**

# 2.Предмет на техническото задание

Система предлага съхраняване на данни за филми, филтриране по определени категории, тяхното извличане, редактиране, както и изтриване . В техническото задание се описват пълните и точни първоначални изисквания към проекта, неговите характеристики, както и изпълняваните от него функции и крайната цел и идея на програмния продукт.

Модул за въвеждане на информация за филм

- възможност за въвеждане на име;

- възможност за въвеждане на актьори;

- възможност за въвеждане на режисьор;

- възможност за въвеждане на година;

- възможност за въвеждане на жанр;

- възможност за въвеждане на рейтинг

Модул за търсене на филми, според определени критерии

- възможност за търсене по име;

- възможност за търсене по жанр;

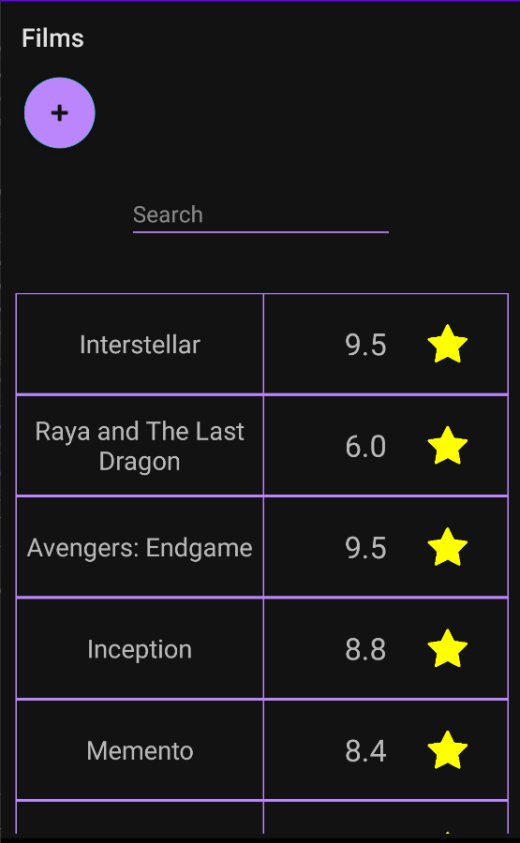
- възможност за търсене по актьори;

- възможност за търсене по година;

- възможност за търсене по режисьор

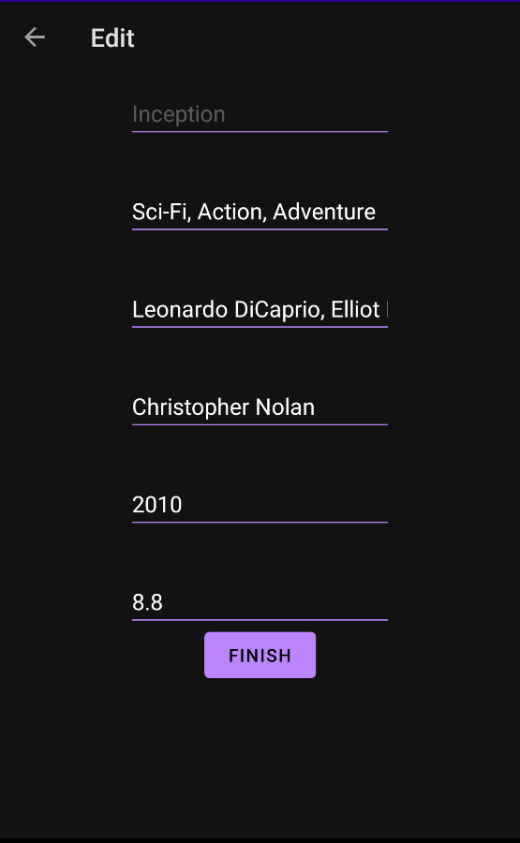
**Функционални изисквания**

*Начален екран*

**

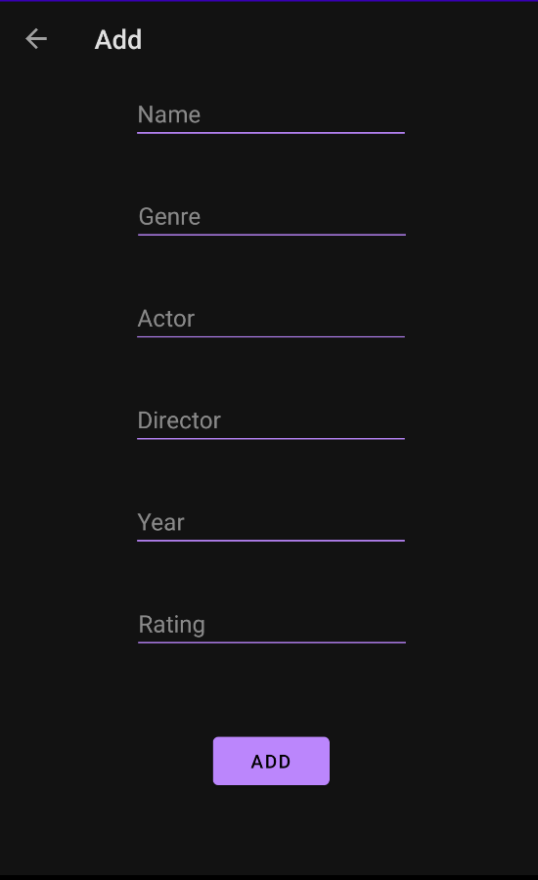
Да се зарежда този екран при влизане в приложението и да съдържа списък със всички въвеждани филми. За всеки филм се извежда име и оценка, дадена му от потребителя. В горния ляв ъгъл да има бутон за добавяне на филм. При избор на филм да се извежда цялата информация за филма.

*Екрани за информация на филм и за редактиране на филм*



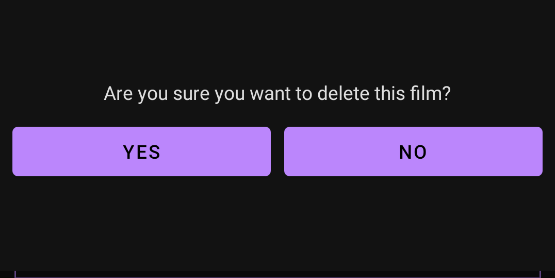
В долния ляв ъгъл на екрана с информацията за филм да има бутон за изтриване на филма, а в долния десен бутон за редакция, който да води към съответния екран. Екрана за корекция на данните да позволява редактиране на всички атрибути на филма без името. Съхранението на редакциите да става едва когато данните са напълно коректни и след натискане на бутон Finish.

*Екран за добавяне на нов филм*



Да съдържа полета за въвеждане на име на филма, жанр, актьори, режисьори, година на издаване и оценка, дадена му от потребителя. За избор на оценката да има скала от 1 до 10. Всички данни да се изпратят до базата от данни едва след проверка за тяхната коректност и натискане на бутон Add, както и да не се позволява въвеждане на име на филм, който вече съществува в нея.

*Екран за потвърждение*



При натискане на бутоните за изтриване, добавяне или редактиране на филм, да се извежда изскачащ прозорец с допитване към потребителя, което потвърждава финалното му решение.

**Нефункционални изисквания**

Софтуерният продукт да може да се изтегли и инсталира на Android мобилни устройства с версия на операционната система 7.0 и по-нова. Да заема минимално място във вътрешната памет на устройството.

Да бъде разработен на език за програмиране Java, поради масовото му използване за програмиране на Android приложения.

Да се използва релационната база от данни SQLite за съхраняване на записи за филми, която е вградена в Android, поради бързината й, и сравнително леките й изисквания от стана на производителност на системата.

# 3.Обхват на разработката

Разработеният софтуерен продукт трябва да удовлетворява очакванията на потребителите и да им предоставя възможност за улеснено използване на заложените в него функционалности. Потребителите ще имат на разположение продукт, който им позволява да съхраняват информация за вече гледани филми, давайки им оценка според удовлетвореността си. Те ще могат лесно да филтрират своя списък от филми според жанр, име на филм, актьор, режисьор, както и година на премиера. Приложението поддържа и изтриване на създаден запис. Всички действия, които се обвързват с промяна на атрибутите на филм в базата данни се правят само след обширна валидация, както и потвърждение от страна на потребителя.

# 4.Система за контрол на версиите

[Система за контрол на версиите](http://bg.wikipedia.org/wiki/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.

**Избор и обосновка**

За разработването на софтуерния продукт е използвана децентрализираната система за контрол на версиите – **GitHub**. GitHub е базирана на Git, като премахва ръчното въвеждане на команди от терминал и го замества с интуитивно десктоп и уеб приложение. Тя е бърза, ефективна и разполага със система за нелинейна разработка. С GitHub много потребители могат да работят едновременно по един и същ файл, без това да доведе до хаос в съдържанието на файла.

**Предимства:**

* скорост;
* опростен дизайн;
* мощна поддръжка за нелинейна разработка;
* напълно разпределена работа.

**Недостатъци:**

* въпреки опростения в сравнение с Git дизайн, GitHub се състои от много команди и опции и е нужно високо ниво на разбиране, тъй като някои команди и аргументи са непоследователни до някаква степен;
* слаби инструменти за бинарни файлове;
* не поддържа празни директории;
* липса на ограничение за контрол на достъпа;

**Трудности при използване**

Главните трудности бяха началното научаване на командите и начина на работа. Не възникнаха конфликти, понеже GitHub преди да изпълни команда push предупреждава за направени предишни промени на онлайн хранилището (при наличие на такива) и дава опцията на потребителя първо да изтегли (pull) промените преди да продължи с push на неговите.

**Настройки**

Настройките, необходими за работа с GitHub са потребителски профили от страна на членовете на екипа, както и тяхното добавяне в общо уеб хранилище. След това е необходимо всеки член да клонира онлайн хранилището на собствения си компютър, създавайки локално хранилище. В локалното хранилище всеки член може независимо от другия да извършва промени в него и на файловете в него, както и взаимно да споделят тези промени.

**Съпоставка с други системи**

Единият от възможните избори бе система за контрол на версиите от първо поколение, например RCS и SCCS. Принципът на работа при такъв тип система е сървърно базирано хранилище и „заключване“ (правейки го достъпен само за четен от не работещ по него разработчик) на достъп до файл, ако член на екипа работи по него в даден момент. Принцип на работа на GitHub като децентрализирана система е различен. При него всеки от екипа може да работи по един и същ файл едновременно, но е в локално хранилище. Въпреки, че при опит за записване на няколко различни копия на този файл на сървърното хранилище може да доведе до конфликт, GitHub решава този проблем като:

* предоставя възможност на своите потребители да разпределят роли на членовете на екипа, ограничавайки достъп до определени файлове
* използва т.н. branches
* предоставя възможност за командите push и pull да бъдат превърнати в push request и pull request за потребители с ограничени права до хранилището

Втората съпоставка е с централизираните системи за контрол на версиите като CVS, Subversion и Perforce. Главната разлика при тях е липсата на локални хранилища, което прави наличието на интернет връзка напълно задължителна по време на работа върху файла на сървърното хранилище. Едни от предимствата на централизираните системи са, че са по-лесни за научаване и по-добрата им работа с бинарни файлове.

# 5.Система за управление на проекта

**Избор и обосновка**

**Trello** е безплатно уеб приложение за управление на проекти, подпомагащ увеличаването на производителността при изпълнение на колективна работа. Trello използва методологията Канбан. To 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 **To Do, Doing, Done**. В тези списъци се поставят задачи – карти. Предоставя и възможност за назначаване на различни потребители към различни задачи, както и тяхното оцветяване за по-лесно ориентиране в бъдеще.

**Предимства:**

* изключително интуитивен софтуер за управление на проекти, както и на ежедневни задачи;
* настройките за начало на работа могат да бъдат напълно минимални, както и по-подробни, в зависимост от проекта
* възможност за попълване на различни данни на видимата предна част на картата и както и на нейният „гръб“
* наличие на много разнообразни шаблони за дъски според нуждите на потребителя

**Недостатъци:**

* простотата и минимализмът могат да причинят затруднения при използването й в по-сложен проект
* налична е само с наличие на връзка към интернет
* няма възможност да се видят задачите, подредени по потребител или краен срок.

**Трудности при използване**

Екипът ни не изпита особени трудности при работа с Trello. Изключително минималистичният му и лек дизайн ни позволи да организираме задачите без забавяния.

**Настройки**

За използване на Trello първо трябва да се избере web, desktop или mobile версия. След това са необходими потребителски профили на членовете на екипа. Следва мениджъра на екипа да създаде workspace, към който да приложи някоя от богатия избор за шаблони, да добави членовете към него, да разпредели техните права в рамките на този workspace и да започне да създава категории и карти със задачи.

# 6.Методология на разработване

**Екстремно програмиране, Extreme Programming, XP**

Методологията на разработване, която сме избрали е Екстремно програмиране, тази методология спада към гъвкавите (agile) методологии.

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. Екстремното програмиране извършва фази в много малки непрекъснати стъпки, което значи, че често има работещи модули на приложението, готови за тестване и поправяне при поява на грешки.

Някои други характеристики на Екстремното програмиране:

* метод на инкрементално планиране(стъпка по стъпка);
* непрекъснато тестване;
* непрекъсната интензивна комуникация;
* тясно сътрудничество м/у програмистите;
* клиентско съдействие измества договорно споразумение
* лесно реагиращ на промяна софтуер вместо стриктното следване на план

**Предимства на XP:**

* Възможност за наблягане върху кода и пълноценната разработка на самото приложение
* Постоянно тестване на модули, довеждащо много често до готов продукт, малко след написването на последната линия код
* Възможност за „слушане“ на променящите се изисквания на клиента и тяхното бързо прилагане към разработвания продукт
* Създаване на добър базисен дизайн в началото на проекта, който след това може да се надгражда

**Недостатъци на XP:**

* Липсата на подробен план в началото може да е объркващо за програмисти, свикнали с някои други методологии като Waterfall
* Крайният срок често се оказва неточен. Той реално зависи от уменията на програмистите, взимайки предвид силното наблягане на методологията върху самата реализация на софтуерното приложение

**Съпоставка на XP със Scrum:**

Екстремното програмиране и Scrum са разбираемо много сходни и подравнени методологии. Има обаче фини, но важни разлики между тези два метода. Scrum спринтовете траят 2-4 седмици, докато типичните XP повторения са по-кратки. Обикновено Scrum екипите не позволяват промени в спринтовете, но XP екипите са малко по-гъвкави към промените в итерациите. Например, след планирането на спринта, наборът от елементи на този спринт остава непроменен, но функция, която не е започнала работа, може по всяко време да бъде заменена с друга функция в XP. Друга разлика между XP и Scrum е, че редът на функции, разработени в XP, е строго приоритизиран от клиента, докато екипът на Scrum решава реда на артикулите (след като изоставането на продукта е приоритетно от собственика на продукта на Scrum).

# 7.Екип, роли и задачи

**Екип**

1. Добромир Начев

2. Александър Стоименов

**Роли**

Добромир Начев – мениджър, разработчик, дизайнер, тестер

Александър Стоименов – автор техническа документация, дизайнер, тестер

**Задачи**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер на задачата | Описание | Продължителност | Предшественик | Изпълнители |
| 1 | **Начало на проекта** | 0 days | - |  |
| 2 | **Анализ на изискванията** | 1 day | - |  |
| 3 | Функционални | 1 day | - | Александър |
| 4 | Нефункционални | 1 day | - | Добромир |
| 5 | **Планиране** | 2 days |  |  |
| 6 | Архитектура | 1 day | 2 | Добромир |
| 7 | База данни | 1 day | 2 | Добромир |
| 8 | Бизнес логика | 1 day | 6, 7 | Александър |
| 9 | GUI | 1 day | 6, 7 | Добромир |
| 10 | **Реализация на приложението** | 6 days |  |  |
| 11 | Разработка на базисна структура | 1 day | 5 | Добромир |
| 12 | Създаване на БД | 1 day | 5 | Добромир |
| 13 | Реализация на бизнес логика | 2 days | 11, 12 | Добромир |
| 14 | GUI | 1 day | 11, 12 | Добромир |
| 15 | Тестване | 4 days | 5 | Добромир |
| 16 | **Документация** | 3 days |  |  |
| 17 | Обща | 2 days | 10 | Александър |
| 18 | Потребителска | 1 day | 17 | Добромир |

**Определяне на срокове**

Начална дата: 24/05/2021

Анализ на изискванията: 24/05/2021

Планиране: 26/05/2021

Разработка: 01/06/2021

Тестване: 01/06/2021

Документиране: 04/06/2020

**Програмни средства за реализация**

* Език за програмиране: Java
* База данни: SQLite
* Емулация на приложението за тестване: Genymotion
* IDE: Android Studio

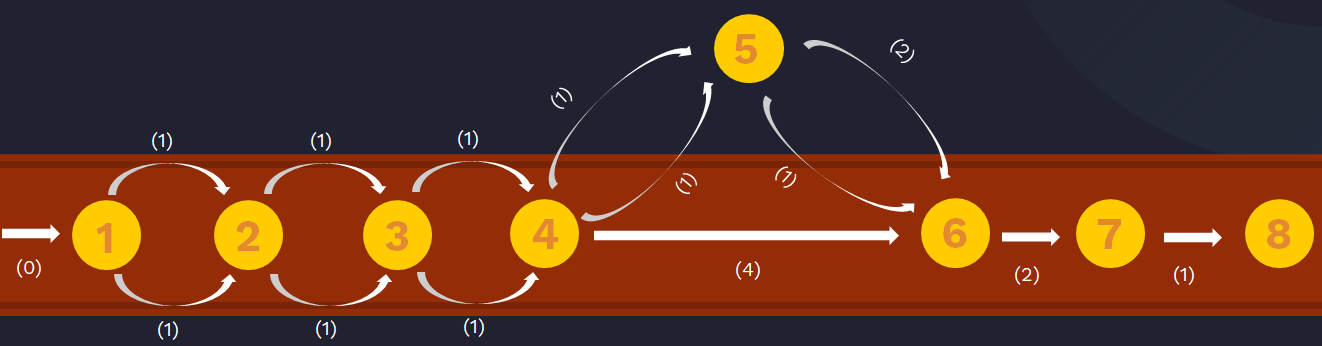
**Стъпки, зависещи от конкретния подход за разработка**

* Краткотрайни итерации по програмиране на приложението с готови за тестване модули в края на всяка
* Междуекипно сътрудничество
* Постоянна готовност за действие при промяна на клиентски изисквания

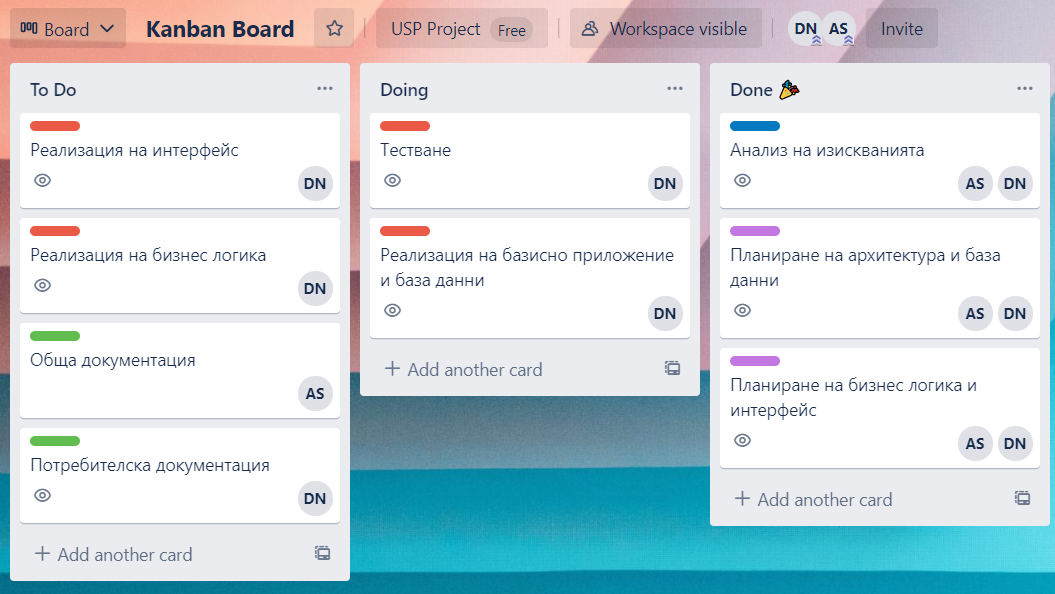
**Жалонни точки**

|  |  |
| --- | --- |
| **Събитие** | **Описание** |
| 1 | Започнат е проекта |
| 2 | Анализирани са изискванията |
| 3 | Завършено е планирането на архитектурата и БД |
| 4 | Завършено е планирането на бизнес логиката и интерфейса |
| 5 | Реализирани са базисна структура и БД |
| 6 | Реализирани са бизнес логиката, интерфейса и е завършило тестването |
| 7 | Написана е общата документация |
| 8 | Написана е потребителската документация |

**Критичен път**

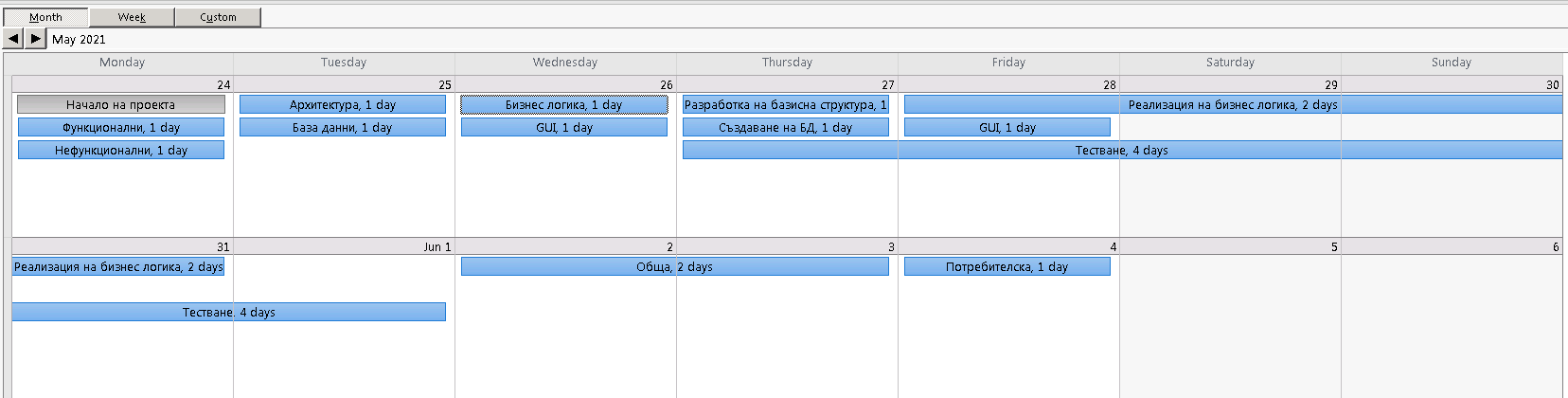


**Trello taskboard**

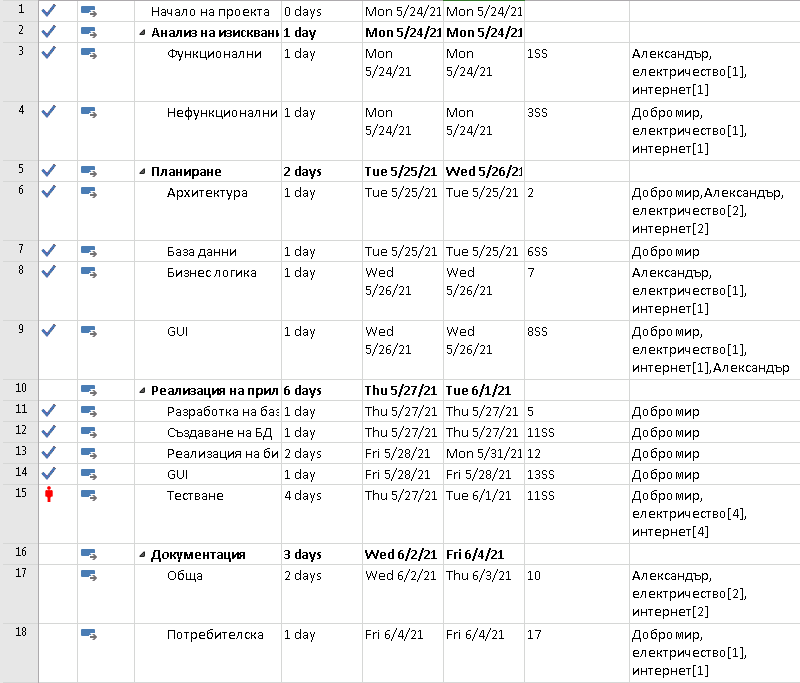


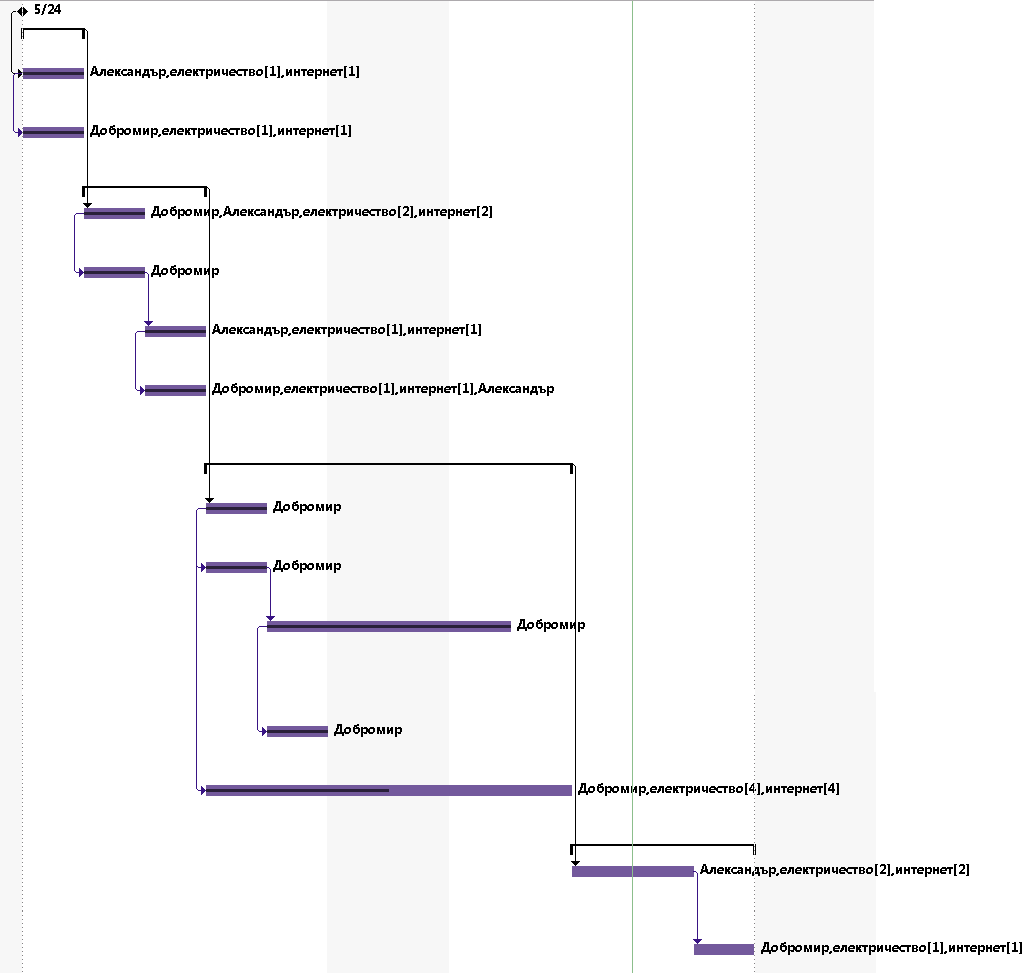
# 8. Microsoft Project

**Календар**

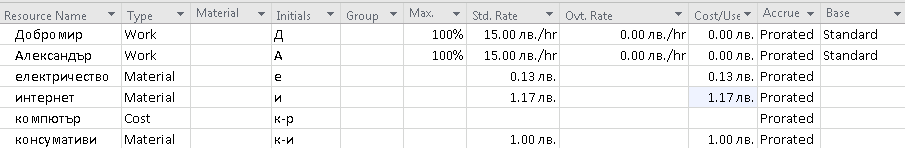


**Диаграма на Гант**





**Ресурсен лист**



***12. Диаграма на свършената работа***

**Диаграма на свършената работа**

**Диаграма и таблица на бюджет**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Fixed Cost | Actual Cost | Remaining Cost | Cost | Baseline Cost | Cost Variance |
| Начало на проекта | 0.00 лв. | 0.00 лв. | 0.00 лв. | 0.00 лв. | 0.00 лв. | 0.00 лв. |
| Анализ на изискванията | 0.00 лв. | 245.20 лв. | 0.00 лв. | 245.20 лв. | 0.00 лв. | 245.20 лв. |
| Планиране | 0.00 лв. | 609.10 лв. | 0.00 лв. | 609.10 лв. | 0.00 лв. | 609.10 лв. |
| Реализация на приложението | 0.00 лв. | 843.90 лв. | 242.60 лв. | 1,086.50 лв. | 0.00 лв. | 1,086.50 лв. |
| Документация | 0.00 лв. | 0.00 лв. | 366.50 лв. | 366.50 лв. | 0.00 лв. | 366.50 лв. |

**Диаграма и таблица на ресурси**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Cost** | **Baseline Cost** | **Cost Variance** |
| Добромир | 1,680.00 лв. | 0.00 лв. | 1,680.00 лв. |
| Александър | 600.00 лв. | 0.00 лв. | 600.00 лв. |

# 9.Бизнес логика

**Бизнес правила**

* 1 филм може да има само едно заглавие
* 1 филм може да има много жанрове
* 1 филм може да има повече от един актьор
* 1 филм може да има повече от един режисьор
* 1 филм може да има само една година на премиера
* 1 филм може да има само един рейтинг

**Процедурни правила**

* Всички полета трябва да са попълнени при записа на нов филм
* Рейтингът не може да е по-малък от 1 и по-голям от 10
* Годината не може да е по-малка от 1888 или по-голяма от 2049

**Обекти и техните атрибути**

Film

* Id – идентификатор
* Name – име
* Genres – жанрове
* Actors – актьори
* Director – режисьори
* Year – година
* Rating – оценка

# 10.Изводи

За да се завърши успешно един проект, удовлетворяващ изискванията на клиента и потребителите трябва да се отдели достатъчно време на всяка отделна фаза от неговата разработка. Трябва внимателно да се извършват изчисленията при създаването на бюджета и да се вземе предвид възможността от настъпване на рискови ситуации, както и отклонение от проектния план.

Trello като система за управление на проекти е много минималистична и лесна за употреба, но е предназначена най-вече за малки екипи и проекти поради своята лимитираност.

Децентрализираните системи за контрол на версии като GitHub позволяват съвместна работа на екипи от всякакви размери, давайки възможности за регулиране на достъп, история на проекта и извършване на експериментални промени по файлове, без да се засягат главните, дори и след push - `чрез branches.