

Diseño de Aplicaciones 2

Obligatorio 1

Evidencia de la aplicación de TDD y Clean Code

Entregado como requisito de la materia Diseño de Aplicaciones 2

2021

Diego Scaffo - 196673
José Pedro Amado - 188885

Índice

1. Descripción de la estrategia de TDD seguida	2
2. Informe de cobertura para todas las pruebas desarrolladas	7
3. Clean Code	8

1. Descripción de la estrategia de TDD seguida

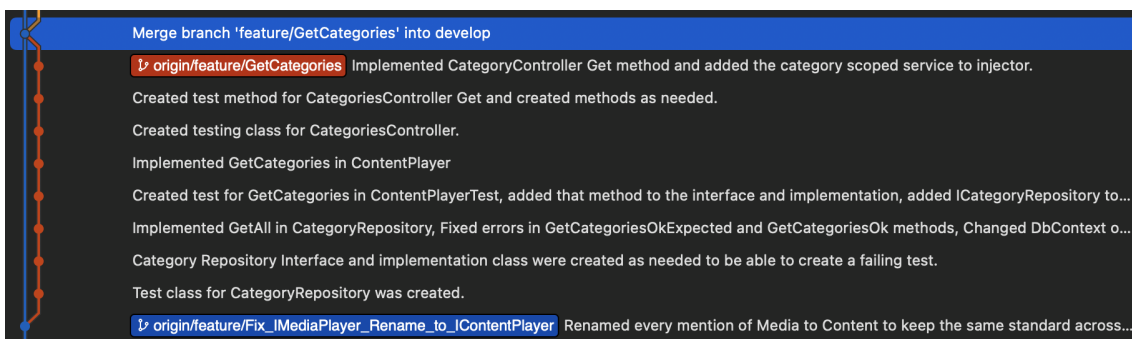
El proyecto se realizó con TDD en casi su totalidad. El enfoque propuesto fue el de Outside In, realizando en primera instancia un diseño del sistema a ser luego desarrollado y refactorizado en base a las pruebas realizadas.

En la práctica se vio que el diseño realizado en primera instancia ayudó a guiar el desarrollo en las etapas más tempranas, pero fue mutando a medida que las pruebas hacían salir a la luz nuevas necesidades para poder implementar las funcionalidades solicitadas.

Se hizo una separación de implementación y testing a nivel de proyectos. Cada proyecto a ser testeado tiene su respectivo paquete para este fin, por ejemplo, BL y BL.Test.

Las siguientes capturas de las distintas ramas del repositorio muestran evidencia del desarrollo de esta técnica para las funcionalidades donde ésta era requerimiento.

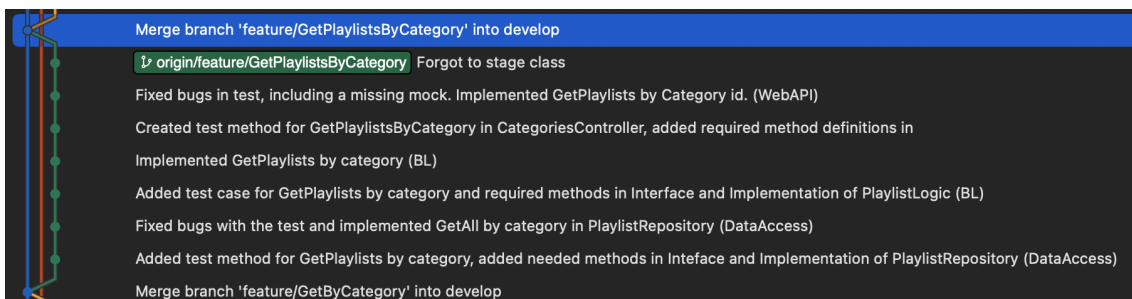
Get Categorías



Merge branch 'feature/GetCategories' into develop

- origin/feature/GetCategories Implemented CategoryController Get method and added the category scoped service to injector.
- Created test method for CategoriesController Get and created methods as needed.
- Created testing class for CategoriesController.
- Implemented GetCategories in ContentPlayer
- Created test for GetCategories in ContentPlayerTest, added that method to the interface and implementation, added ICategoryRepository to...
- Implemented GetAll in CategoryRepository, Fixed errors in GetCategoriesOkExpected and GetCategoriesOk methods, Changed DbContext o...
- Category Repository Interface and implementation class were created as needed to be able to create a failing test.
- Test class for CategoryRepository was created.
- origin/feature/Fix_ILMediaPlayer_Rename_to_IContentPlayer Renamed every mention of Media to Content to keep the same standard across...

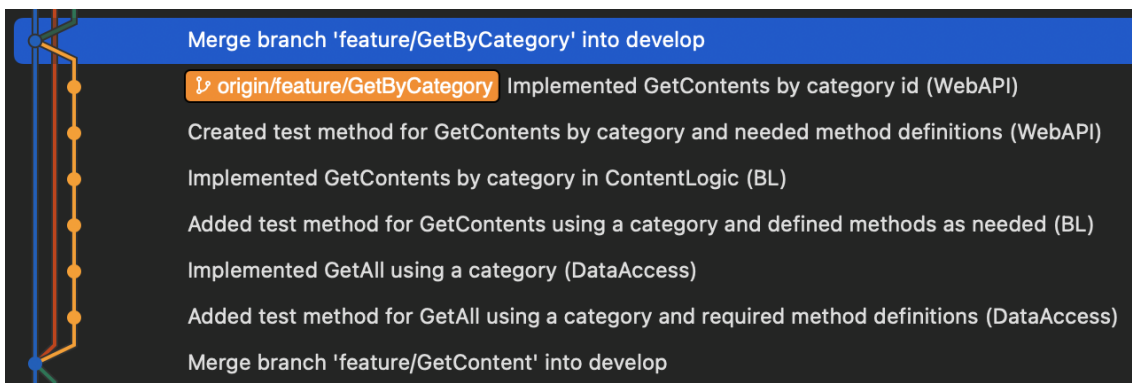
Get Playlist de Categoría



Merge branch 'feature/GetPlaylistsByCategory' into develop

- origin/feature/GetPlaylistsByCategory Forgot to stage class
- Fixed bugs in test, including a missing mock. Implemented GetPlaylists by Category id. (WebAPI)
- Created test method for GetPlaylistsByCategory in CategoriesController, added required method definitions in
- Implemented GetPlaylists by category (BL)
- Added test case for GetPlaylists by category and required methods in Interface and Implementation of PlaylistLogic (BL)
- Fixed bugs with the test and implemented GetAll by category in PlaylistRepository (DataAccess)
- Added test method for GetPlaylists by category, added needed methods in Interface and Implementation of PlaylistRepository (DataAccess)
- Merge branch 'feature/GetByCategory' into develop

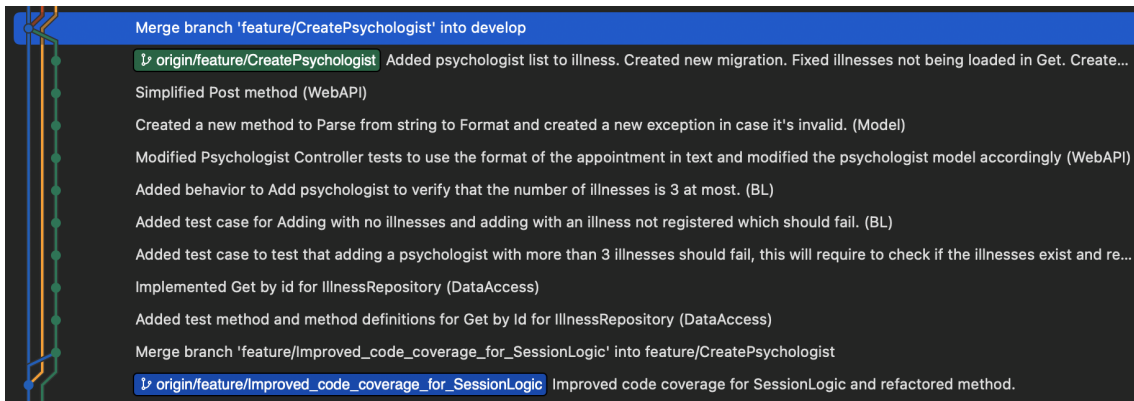
Get Contenidos por Categoría



Merge branch 'feature/GetByCategory' into develop

- origin/feature/GetByCategory Implemented GetContents by category id (WebAPI)
- Created test method for GetContents by category and needed method definitions (WebAPI)
- Implemented GetContents by category in ContentLogic (BL)
- Added test method for GetContents using a category and defined methods as needed (BL)
- Implemented GetAll using a category (DataAccess)
- Added test method for GetAll using a category and required method definitions (DataAccess)
- Merge branch 'feature/GetContent' into develop

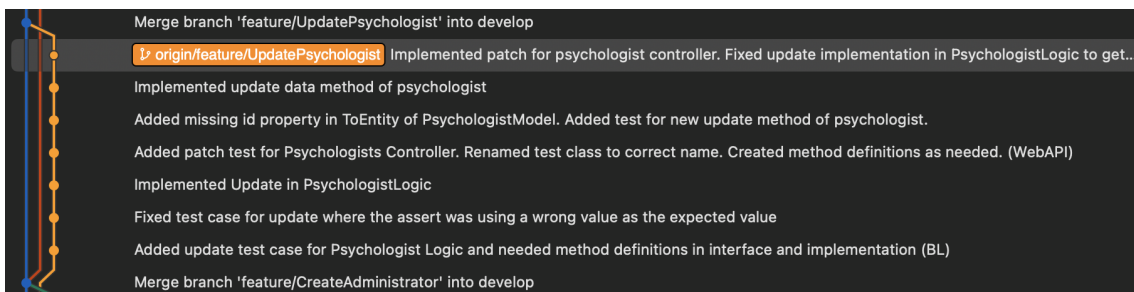
Alta Psicólogo



A screenshot of a Git commit history window. The left sidebar shows a vertical timeline of commits with colored markers. The main area lists the following commits from top to bottom:

- Merge branch 'feature/CreatePsychologist' into develop
- origin/feature/CreatePsychologist** Added psychologist list to illness. Created new migration. Fixed illnesses not being loaded in Get. Create... Simplified Post method (WebAPI)
- Created a new method to Parse from string to Format and created a new exception in case it's invalid. (Model)
- Modified Psychologist Controller tests to use the format of the appointment in text and modified the psychologist model accordingly (WebAPI)
- Added behavior to Add psychologist to verify that the number of illnesses is 3 at most. (BL)
- Added test case for Adding with no illnesses and adding with an illness not registered which should fail. (BL)
- Added test case to test that adding a psychologist with more than 3 illnesses should fail, this will require to check if the illnesses exist and re...
- Implemented Get by id for IllnessRepository (DataAccess)
- Added test method and method definitions for Get by Id for IllnessRepository (DataAccess)
- Merge branch 'feature/Improved_code_coverage_for_SessionLogic' into feature/CreatePsychologist
- origin/feature/Improved_code_coverage_for_SessionLogic** Improved code coverage for SessionLogic and refactored method.

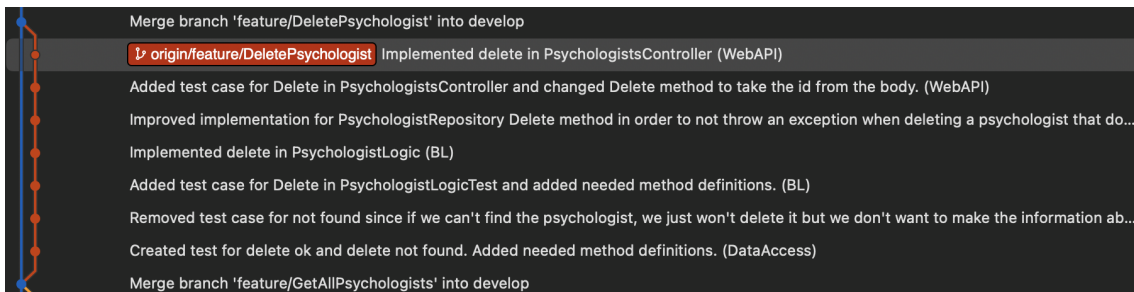
Modificación Psicólogo



A screenshot of a Git commit history window. The left sidebar shows a vertical timeline of commits with colored markers. The main area lists the following commits from top to bottom:

- Merge branch 'feature/UpdatePsychologist' into develop
- origin/feature/UpdatePsychologist** Implemented patch for psychologist controller. Fixed update implementation in PsychologistLogic to get...
- Implemented update data method of psychologist
- Added missing id property in ToEntity of PsychologistModel. Added test for new update method of psychologist.
- Added patch test for Psychologists Controller. Renamed test class to correct name. Created method definitions as needed. (WebAPI)
- Implemented Update in PsychologistLogic
- Fixed test case for update where the assert was using a wrong value as the expected value
- Added update test case for Psychologist Logic and needed method definitions in interface and implementation (BL)
- Merge branch 'feature/CreateAdministrator' into develop

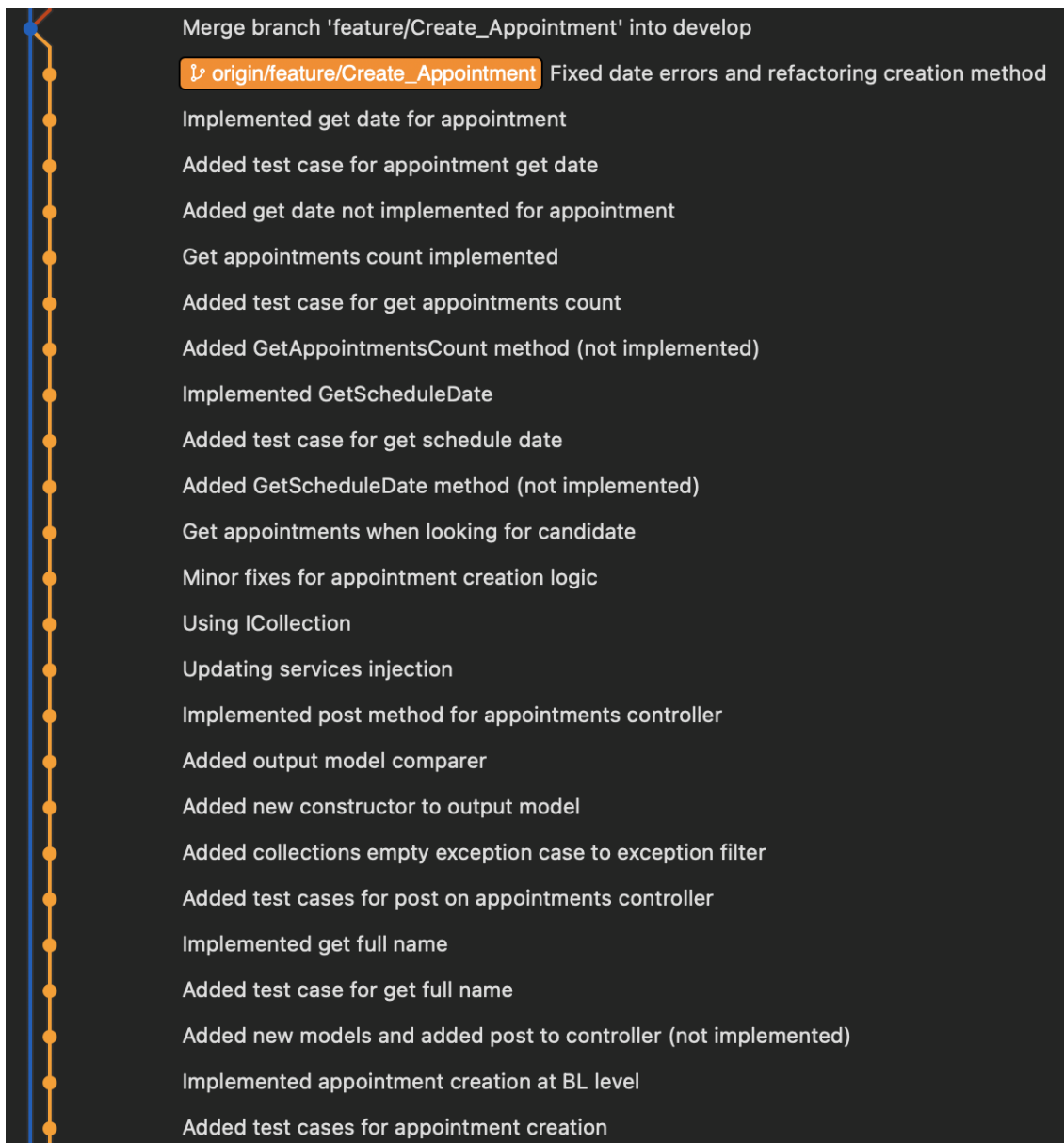
Baja Psicólogo



A screenshot of a Git commit history window. The left sidebar shows a vertical timeline of commits with colored markers. The main area lists the following commits from top to bottom:

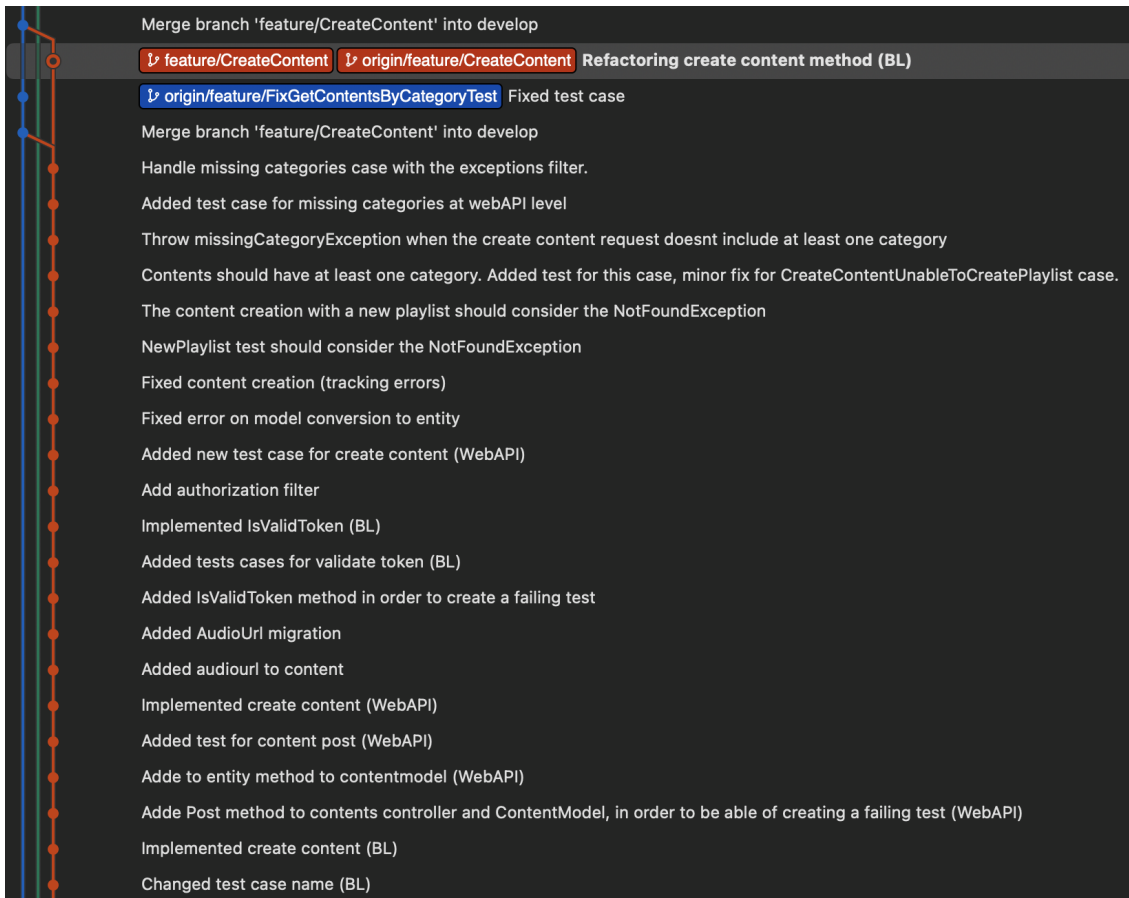
- Merge branch 'feature/DeletePsychologist' into develop
- origin/feature/DeletePsychologist** Implemented delete in PsychologistsController (WebAPI)
- Added test case for Delete in PsychologistsController and changed Delete method to take the id from the body. (WebAPI)
- Improved implementation for PsychologistRepository Delete method in order to not throw an exception when deleting a psychologist that do...
- Implemented delete in PsychologistLogic (BL)
- Added test case for Delete in PsychologistLogicTest and added needed method definitions. (BL)
- Removed test case for not found since if we can't find the psychologist, we just won't delete it but we don't want to make the information ab...
- Created test for delete ok and delete not found. Added needed method definitions. (DataAccess)
- Merge branch 'feature/GetAllPsychologists' into develop

Crear consulta con Psicólogo



```
Merge branch 'feature/Create_Appointment' into develop
origin/feature/Create_Appointment Fixed date errors and refactoring creation method
Implemented get date for appointment
Added test case for appointment get date
Added get date not implemented for appointment
Get appointments count implemented
Added test case for get appointments count
Added GetAppointmentsCount method (not implemented)
Implemented GetScheduleDate
Added test case for get schedule date
Added GetScheduleDate method (not implemented)
Get appointments when looking for candidate
Minor fixes for appointment creation logic
Using ICollection
Updating services injection
Implemented post method for appointments controller
Added output model comparer
Added new constructor to output model
Added collections empty exception case to exception filter
Added test cases for post on appointments controller
Implemented get full name
Added test case for get full name
Added new models and added post to controller (not implemented)
Implemented appointment creation at BL level
Added test cases for appointment creation
```

Alta Contenido



Merge branch 'feature/CreateContent' into develop

[↗ origin/feature/CreateContent](#) Refactoring create content method (BL)

[↗ origin/feature/FixGetContentsByCategoryTest](#) Fixed test case

Merge branch 'feature/CreateContent' into develop

Handle missing categories case with the exceptions filter.

Added test case for missing categories at webAPI level

Throw missingCategoryException when the create content request doesnt include at least one category

Contents should have at least one category. Added test for this case, minor fix for CreateContentUnableToCreatePlaylist case.

The content creation with a new playlist should consider the NotFoundException

NewPlaylist test should consider the NotFoundException

Fixed content creation (tracking errors)

Fixed error on model conversion to entity

Added new test case for create content (WebAPI)

Add authorization filter

Implemented IsValidToken (BL)

Added tests cases for validate token (BL)

Added IsValidToken method in order to create a failing test

Added AudioUrl migration

Added audiourl to content

Implemented create content (WebAPI)

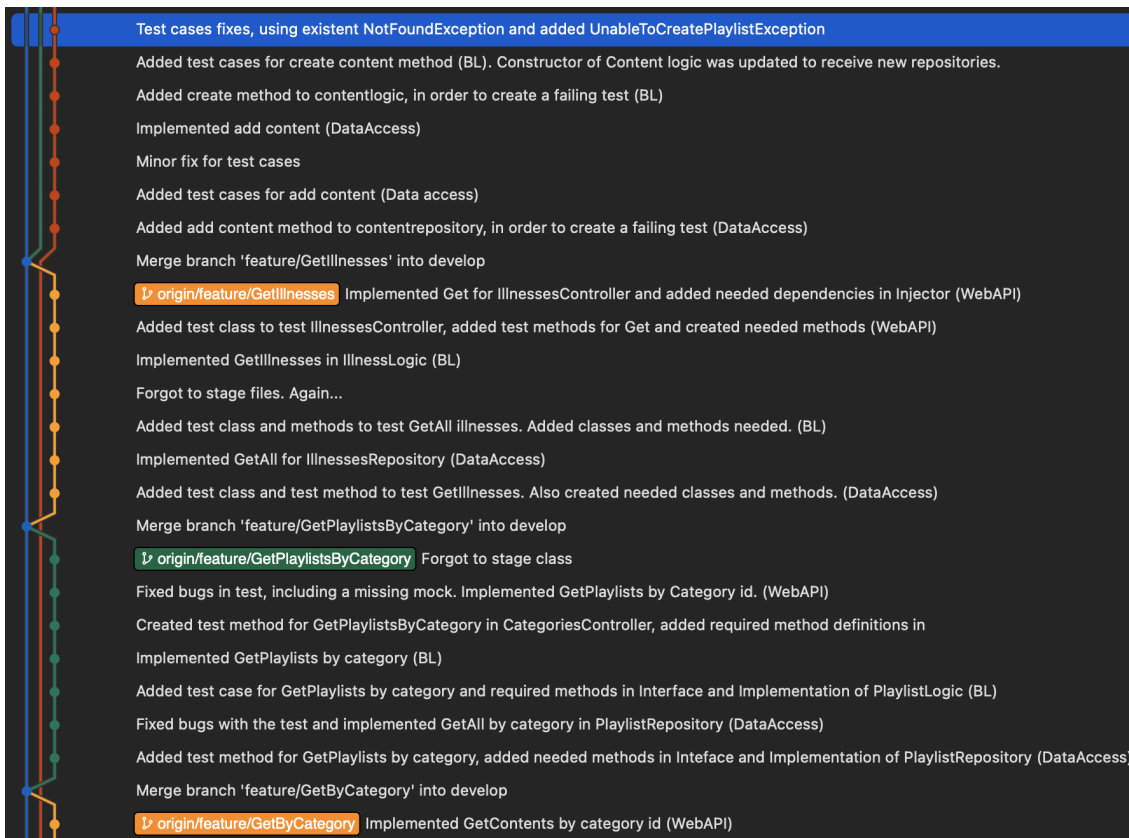
Added test for content post (WebAPI)

Adde to entity method to contentmodel (WebAPI)

Adde Post method to contents controller and ContentModel, in order to be able of creating a failing test (WebAPI)

Implemented create content (BL)

Changed test case name (BL)



Test cases fixes, using existent NotFoundException and added UnableToCreatePlaylistException

Added test cases for create content method (BL). Constructor of Content logic was updated to receive new repositories.

Added create method to contentlogic, in order to create a failing test (BL)

Implemented add content (DataAccess)

Minor fix for test cases

Added test cases for add content (Data access)

Added add content method to contentrepository, in order to create a failing test (DataAccess)

Merge branch 'feature/GetIllnesses' into develop

[↗ origin/feature/GetIllnesses](#) Implemented Get for IllnessesController and added needed dependencies in Injector (WebAPI)

Added test class to test IllnessesController, added test methods for Get and created needed methods (WebAPI)

Implemented GetIllnesses in IllnessLogic (BL)

Forgot to stage files. Again...

Added test class and methods to test GetAll illnesses. Added classes and methods needed. (BL)

Implemented GetAll for IllnessesRepository (DataAccess)

Added test class and test method to test GetIllnesses. Also created needed classes and methods. (DataAccess)

Merge branch 'feature/GetPlaylistsByCategory' into develop

[↗ origin/feature/GetPlaylistsByCategory](#) Forgot to stage class

Fixed bugs in test, including a missing mock. Implemented GetPlaylists by Category id. (WebAPI)

Created test method for GetPlaylistsByCategory in CategoriesController, added required method definitions in

Implemented GetPlaylists by category (BL)

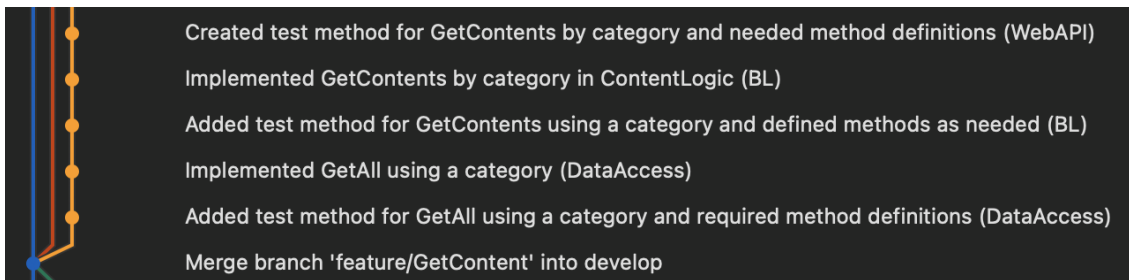
Added test case for GetPlaylists by category and required methods in Interface and Implementation of PlaylistLogic (BL)

Fixed bugs with the test and implemented GetAll by category in PlaylistRepository (DataAccess)

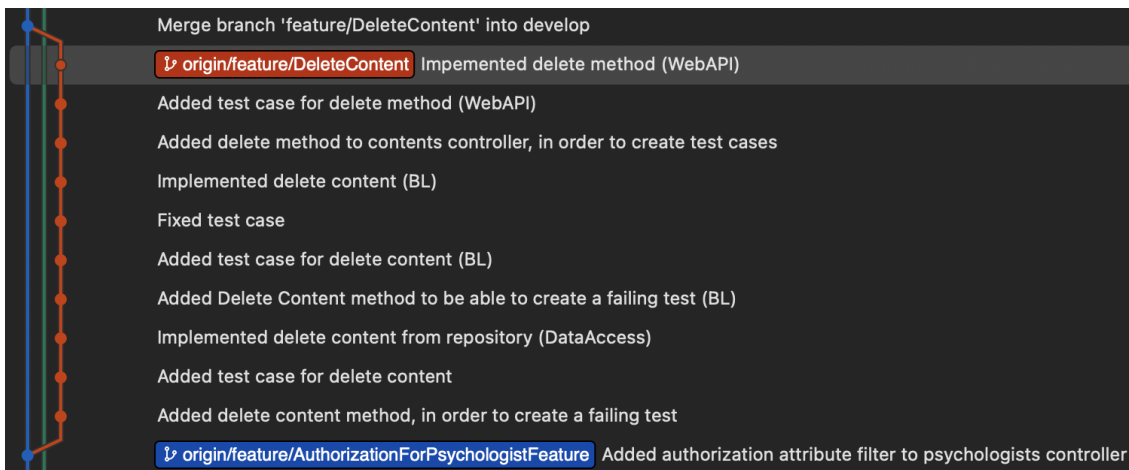
Added test method for GetPlaylists by category, added needed methods in Inteface and Implementation of PlaylistRepository (DataAccess)

Merge branch 'feature/GetByCategory' into develop

[↗ origin/feature/GetByCategory](#) Implemented GetContents by category id (WebAPI)



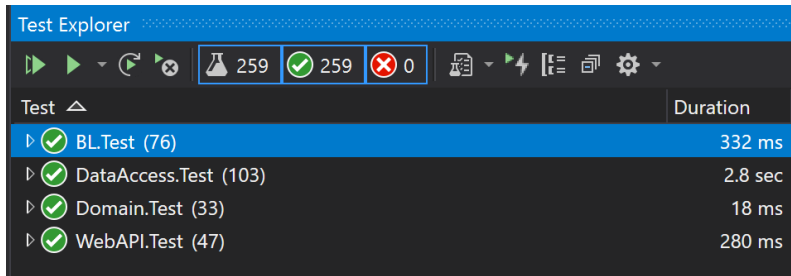
Baja Contenido



2. Informe de cobertura para todas las pruebas desarrolladas

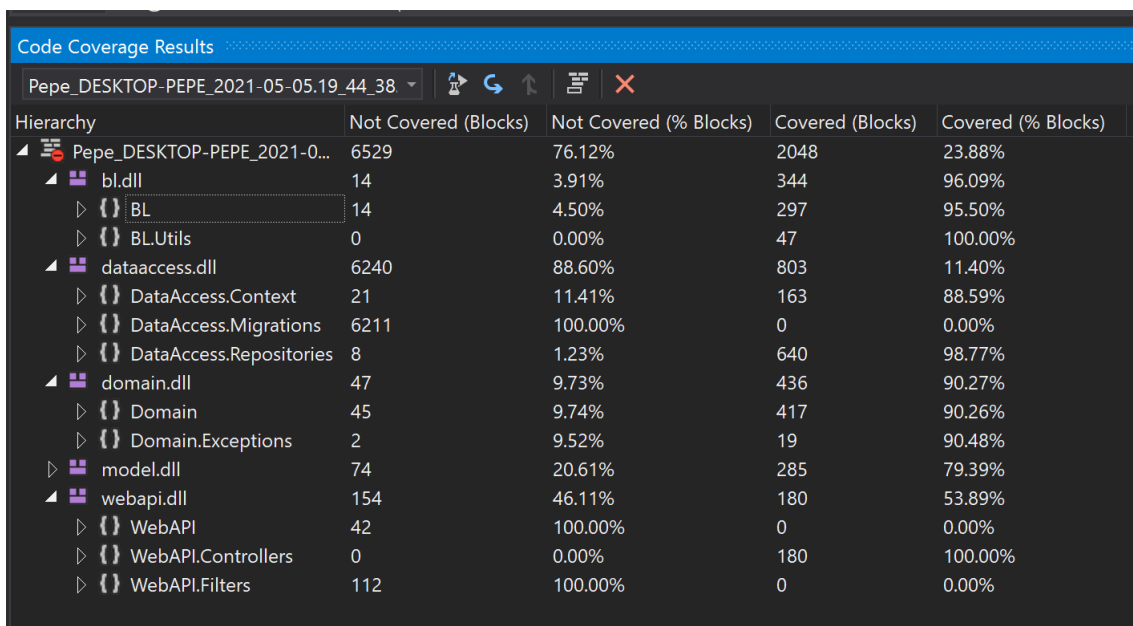
Se tuvo especial cuidado sobre el nivel de cobertura, especialmente sobre los proyectos clave.

Podemos ver en la siguiente captura como están distribuidos los casos de prueba a través de los diferentes proyectos, alcanzando un total de 259 test cases.



Test	Duration
BL.Test (76)	332 ms
DataAccess.Test (103)	2.8 sec
Domain.Test (33)	18 ms
WebAPI.Test (47)	280 ms

A continuación se muestra una captura de los resultados de Code Coverage. Si consideramos los paquetes relevantes, el promedio de porcentaje es de 92.87%.



Hierarchy	Not Covered (Blocks)	Not Covered (% Blocks)	Covered (Blocks)	Covered (% Blocks)
Pepe_DESKTOP-PEPE_2021-05-19_44_38	6529	76.12%	2048	23.88%
bl.dll	14	3.91%	344	96.09%
BL	14	4.50%	297	95.50%
BL.Utils	0	0.00%	47	100.00%
dataaccess.dll	6240	88.60%	803	11.40%
DataAccess.Context	21	11.41%	163	88.59%
DataAccess.Migrations	6211	100.00%	0	0.00%
DataAccess.Repositories	8	1.23%	640	98.77%
domain.dll	47	9.73%	436	90.27%
Domain	45	9.74%	417	90.26%
Domain.Exceptions	2	9.52%	19	90.48%
model.dll	74	20.61%	285	79.39%
webapi.dll	154	46.11%	180	53.89%
WebAPI	42	100.00%	0	0.00%
WebAPI.Controllers	0	0.00%	180	100.00%
WebAPI.Filters	112	100.00%	0	0.00%

3. Clean Code

Respetamos los principios de Clean Code, utilizando nombres descriptivos, evitando abreviaciones, recibiendo pocos parámetros en los métodos, escribiendo métodos cortos, con nombres que impliquen acciones y sean claros. A su vez, se evita utilizar acciones para los nombres de propiedades, se evitan los prefijos y las codificaciones innecesarias.

A continuación, se muestran algunas capturas del código escrito:

```
public class Appointment
{
    public int Id { get; set; }

    public DateTime Date { get; set; }

    public Patient Patient { get; set; }

    public Psychologist Psychologist { get; set; }

    public Illness Illness { get; set; }

    public string Address { get; set; }

    public DateTime GetDate()
    {
        return Date.Date;
    }
}
```

```
[HttpGet]
1 reference
public IActionResult Get()
{
    IEnumerable<Category> categories = this.categoryLogic.GetCategories();
    return Ok(categories);
}
```

```
18 references
public void CreateContent(Content content)
{
    if (content.Categories == null || content.Categories.Count() == 0)
        throw new MissingCategoriesException();

    content.Categories = GetStoredCategories(content.Categories);
    content.PlayLists = GetStoredPlaylists(content.PlayLists);

    this.contentRepository.Add(content);
}
```

5 references

```
public static DateTime CalculateAppointmentDate(Psychologist candidate, int limitOfAppointmentsPerDay)
{
    DateTime today = DateTime.Now.Date;
    Schedule last = candidate.GetLast();
    if (last != null)
    {
        if (last.GetScheduleDate() <= DateTime.Now.Date)
        {
            return CalculateNextWorkDay(today.AddDays(1));
        }
        if (last.Appointments.Count() < limitOfAppointmentsPerDay)
        {
            return last.GetScheduleDate();
        }
        return CalculateNextWorkDay(last.GetScheduleDate().AddDays(1));
    }
    return CalculateNextWorkDay(today.AddDays(1));
}
```