My Project

Generated by Doxygen 1.8.17

1 Assignment 08-09

1.1 Week 8

- Implement at least requirements **1**, **2** and **3**.

1.2 Week 9

- Implement all requirements
- Requirements **4** and **5** must be implemented using inheritance and polymorphism.

1.3 Bonus possibility (0.2p, deadline week 9)

In addition to the file-based implementation for the repository, implement a true database-backed repository. For this, use inheritance and polymorphism. You are free to choose any type of database management system (e.g. `MySQL`, `SQLite`, `PostgreSQL`, `Couchbase` etc.).

1.4 Problem Statement

For your solution to the previous assignment (Assignment 05-06), add the following features:

1. Replace the templated DynamicVector with the STL vector. Use STL algorithms wherever possible in your application (e.g. in your filter function you could use `copy_if`, `count_if`). Replace all your for loops either with STL algorithms, or with C++11's ranged-based for loop.
2. Store data in a text file. When the program starts, entities are read from the file. Modifications made during program execution are stored in the file. Implement this using the `iostream` library. Create insertion and extraction operators for your entities and use these when reading/writing to files or the console.
3. Use exceptions to signal errors:
   - from the repository;
   - validation errors – validate your entities using Validator classes;
   - create your own exception classes.
   - validate program input.
4. Depending on your assignment, store your (adoption list, movie watch list, shopping basket or tutorial watch list) in a file. When the application starts, the user should choose the type of file between `CSV` or `HTML`. Depending on the type, the application will save the list in the correct format.

   Indications\ The CSV file will contain each entity on one line and the attributes will be separated by comma \ The HTML file will contain a table, in which each row holds the data of one entity. The columns of the table will contain the names of the data attributes.\ These are exemplified in the `example.csv` and `example.html` files. `CSV` and `HTML` files are used to save and display data to the user; they act as program outputs, so data should not be read from them!
5. Add a new command, allowing the user to see the:
   - adoption list
   - movie watch list
   - shopping basket
   - tutorial watch list\ Displaying the list means opening the saved file (`CSV` or `HTML`) with the correct application (`CSV` files using Notepad, Notepad++, Microsoft Excel etc. and `HTML` files using a browser)
6. Create a UML class diagram for your entire application. For this, you can use any tool that you like (`StarUML` or `LucidChart` are only some examples. Many other options exist.

2 Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

3 Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

4 File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

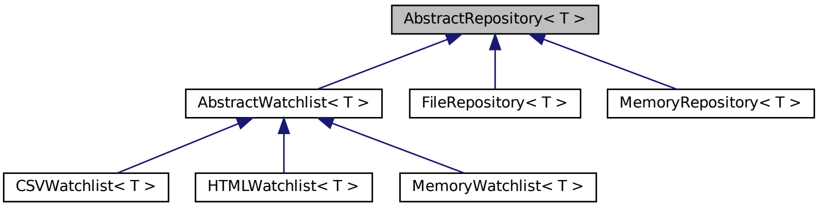
5 Class Documentation

5.1 AbstractRepository< T > Class Template Reference

this is the model for a general purpose repository

`#include <AbstractRepository.h>`

Inheritance diagram for AbstractRepository< T >:



Public Member Functions

- AbstractRepository ()

  the vector that stores the elements of the repository
- virtual ~AbstractRepository ()

  the default destructor of the repository
- virtual void add (const T &elem)=0

  the add method of the repository. Adds an element to the repository
- virtual void update (const T &elem)=0

  the update method of the repository. Updates an element of the repository
- virtual void remove (const T &elem)=0

  removes an element using the id provided by an element with the same id.
- virtual T find (const T &elem)=0

  returns the element with the same id as the element that is provided
- virtual std::vector< T > get_all ()=0

  gets all elements from the repository

5.1.1 Detailed Description

template<class T>
class AbstractRepository< T >

this is the model for a general purpose repository

5.1.2 Constructor & Destructor Documentation

5.1.2.1 AbstractRepository() template<class T >

```
AbstractRepository< T >::AbstractRepository ( )  [inline]
```

the vector that stores the elements of the repository

the default constructor of the repository

5.1.3 Member Function Documentation

5.1.3.1 add() template<class T >

```
virtual void AbstractRepository< T >::add (
             const T & elem )  [pure virtual]
```

the add method of the repository. Adds an element to the repository

Parameters

elem	the element to be added

Exceptions

RepositoryException	if id is taken

Implemented in FileRepository< T >, CSVWatchlist< T >, HTMLWatchlist< T >, MemoryRepository< T >, MemoryWatchlist< T >, FileRepository< Movie >, CSVWatchlist< Movie >, HTMLWatchlist< Movie >, MemoryRepository< Movie >, and MemoryWatchlist< Movie >.

5.1.3.2 find() template<class T >

```
virtual T AbstractRepository< T >::find (
             const T & elem )  [pure virtual]
```

returns the element with the same id as the element that is provided

Parameters

elem	the element used to search the id

Returns

the element from the repository with that id

Exceptions

RepositoryException	if id doesn't exist

Implemented in FileRepository< T >, CSVWatchlist< T >, HTMLWatchlist< T >, MemoryRepository< T >, MemoryWatchlist< T >, FileRepository< Movie >, CSVWatchlist< Movie >, HTMLWatchlist< Movie >, MemoryRepository< Movie >, and MemoryWatchlist< Movie >.

5.1.3.3 get_all() `template<class T >`

```
virtual std::vector<T> AbstractRepository< T >::get_all ( )  [pure virtual]
```

gets all elements from the repository

Returns

all elements currently held in the repository

Implemented in FileRepository< T >, FileRepository< Movie >, CSVWatchlist< T >, HTMLWatchlist< T >, MemoryRepository< T >, CSVWatchlist< Movie >, HTMLWatchlist< Movie >, MemoryRepository< Movie >, MemoryWatchlist< T >, and MemoryWatchlist< Movie >.

5.1.3.4 remove() `template<class T >`

```
virtual void AbstractRepository< T >::remove (
                const T & elem )  [pure virtual]
```

removes an element using the id provided by an element with the same id.

Parameters

elem	the element to be removed

Exceptions

RepositoryException	if id doesn't exist

Implemented in FileRepository< T >, CSVWatchlist< T >, HTMLWatchlist< T >, MemoryRepository< T >, MemoryWatchlist< T >, FileRepository< Movie >, CSVWatchlist< Movie >, HTMLWatchlist< Movie >, MemoryRepository< Movie >, and MemoryWatchlist< Movie >.

5.1.3.5 update() `template<class T >`

```
virtual void AbstractRepository< T >::update (
                const T & elem )  [pure virtual]
```

the update method of the repository. Updates an element of the repository

Parameters

elem	an element with which to update. It should have the same id as the one to be updated

Implemented in FileRepository< T >, CSVWatchlist< T >, HTMLWatchlist< T >, MemoryRepository< T >, MemoryWatchlist< T >, FileRepository< Movie >, CSVWatchlist< Movie >, HTMLWatchlist< Movie >, MemoryRepository< Movie >, and MemoryWatchlist< Movie >.

The documentation for this class was generated from the following file:

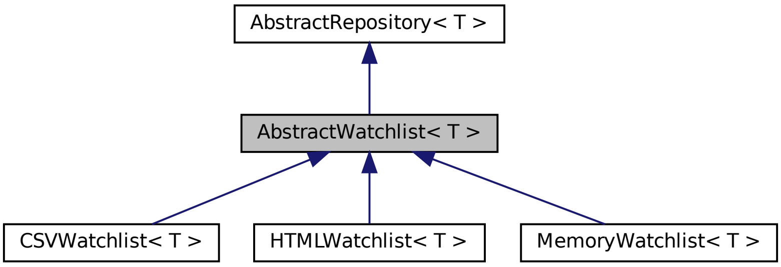
- include/repository/AbstractRepository.h

5.2 AbstractWatchlist< T > Class Template Reference

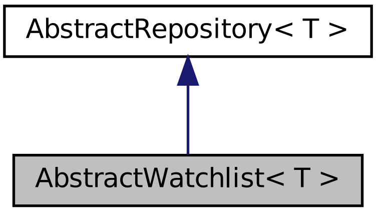
this struct contains the model for an AbstractWatchlist

```
#include <AbstractWatchlist.h>
```

Inheritance diagram for AbstractWatchlist< T >:



Collaboration diagram for AbstractWatchlist< T >:



Public Member Functions

- virtual void save () const =0

  This method should save to a file.
- virtual void display () const =0

  This method display the file.

5.2.1 Detailed Description

template<**class T**>
class AbstractWatchlist< **T** >

this struct contains the model for an AbstractWatchlist

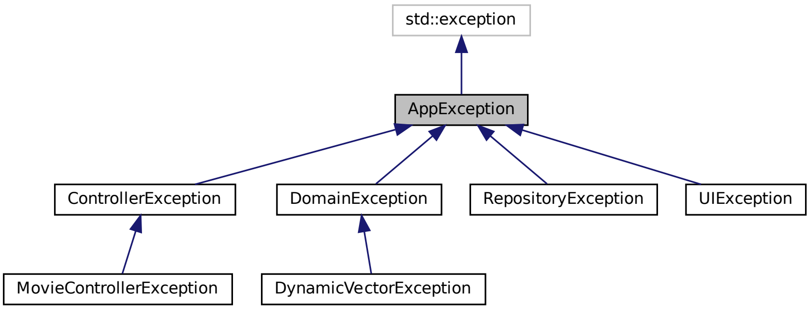
The documentation for this class was generated from the following file:

- include/repository/AbstractWatchlist.h

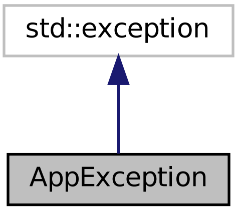
5.3 AppException Class Reference

```
#include <Exceptions.h>
```

Inheritance diagram for AppException:



Collaboration diagram for AppException:



Public Member Functions

- **AppException** (const std::string &message)
- std::string **get_message** () const

5.3.1 Detailed Description

The base app exception.

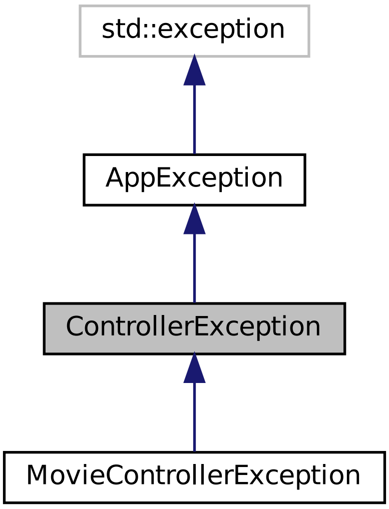
The documentation for this class was generated from the following files:

- include/domain/Exceptions.h
- src/domain/Exceptions.cpp

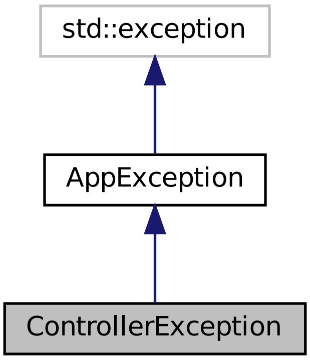
5.4 ControllerException Class Reference

```
#include <Exceptions.h>
```

Inheritance diagram for ControllerException:



Collaboration diagram for ControllerException:



Public Member Functions

- **ControllerException** (const std::string &message)

5.4.1 Detailed Description

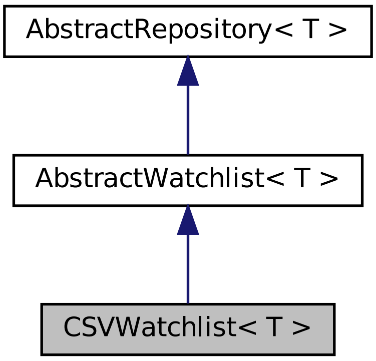
Controller Exception

The documentation for this class was generated from the following files:

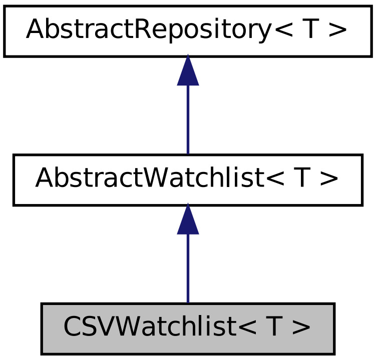
- include/domain/Exceptions.h
- src/domain/Exceptions.cpp

5.5 CSVWatchlist< T > Class Template Reference

Inheritance diagram for CSVWatchlist< T >:



Collaboration diagram for CSVWatchlist< T >:



Public Member Functions

- CSVWatchlist (const std::string &file_path)

  the default constructor of the repository Initializes a repository which holds an empty vector of type T
- ∼CSVWatchlist ()

  the default destructor of the repository
- void add (const T &elem) override

  the add method of the repository. Adds an element to the repository
- void update (const T &elem) override

  the update method of the repository. Updates an element of the repository
- void remove (const T &elem) override

  removes an element using the id provided by an element with the same id.
- T find (const T &elem) override

  returns the element with the same id as the element that is provided
- std::vector< T > get_all () override

  gets all elements from the repository
- void save () const override
- void display () const override

5.5.1 Constructor & Destructor Documentation

5.5.1.1 CSVWatchlist() `template<class T >`

```
CSVWatchlist< T >::CSVWatchlist (
            const std::string & file_path )
```

the default constructor of the repository Initializes a repository which holds an empty vector of type T

Parameters

file_path	this is the path to the file in which to store the data

5.5.2 Member Function Documentation

5.5.2.1 add()

```
template<class T >
void CSVWatchlist< T >::add (
             const T & elem )  [override], [virtual]
```

the add method of the repository. Adds an element to the repository

Parameters

elem	the element to be added

Exceptions

RepositoryException	if its id is already taken

Implements AbstractRepository< T >.

5.5.2.2 display()

```
template<class T >
void CSVWatchlist< T >::display  [override], [virtual]
```

This file shows the file

Implements AbstractWatchlist< T >.

5.5.2.3 find()

```
template<class T >
T CSVWatchlist< T >::find (
             const T & elem )  [override], [virtual]
```

returns the element with the same id as the element that is provided

Parameters

elem	the element used to search the id

Returns

the element from the repository with that id

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.5.2.4 get_all() `template<class T >`

```
vector< T > CSVWatchlist< T >::get_all  [override], [virtual]
```

gets all elements from the repository

Returns

all elements currently held in the repository

Implements AbstractRepository< T >.

5.5.2.5 remove() `template<class T >`

```
void CSVWatchlist< T >::remove (
             const T & elem )  [override], [virtual]
```

removes an element using the id provided by an element with the same id.

Parameters

elem	the element to be removed

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.5.2.6 save() `template<class T >`

```
void CSVWatchlist< T >::save  [override], [virtual]
```

This function saves the data to a file

Implements AbstractWatchlist< T >.

5.5.2.7 update() `template<class T >`

```
void CSVWatchlist< T >::update (
             const T & elem )  [override], [virtual]
```

the update method of the repository. Updates an element of the repository

Parameters

elem	an element with which to update. It should have the same id as the one to be updated

Exceptions

RepositoryException	if id doesn't exist

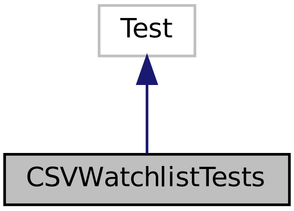
Implements AbstractRepository< T >.

The documentation for this class was generated from the following files:

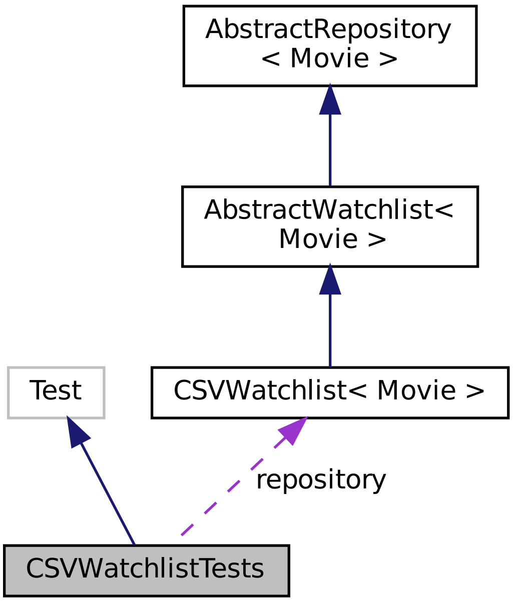
- include/repository/CSVWatchlist.h
- src/repository/CSVWatchlist.cpp

5.6 CSVWatchlistTests Class Reference

Inheritance diagram for CSVWatchlistTests:



Collaboration diagram for CSVWatchlistTests:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

- CSVWatchlist< Movie > **repository**

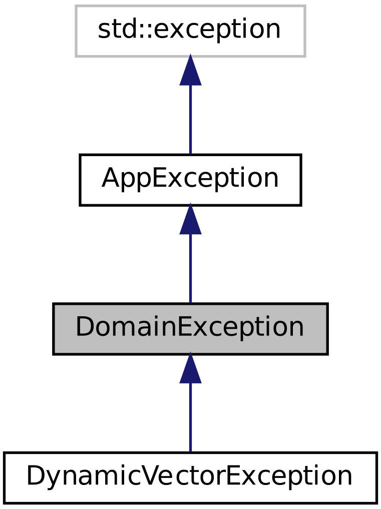
The documentation for this class was generated from the following file:

- tests/repository/CSVWatchlistTests.cpp

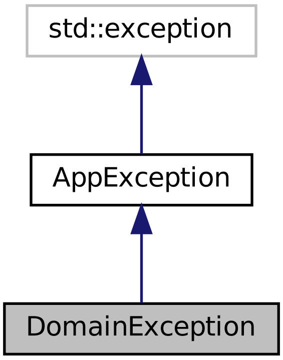
5.7 DomainException Class Reference

```
#include <Exceptions.h>
```

Inheritance diagram for DomainException:



Collaboration diagram for DomainException:



Public Member Functions

- **DomainException** (const std::string &message)

5.7.1 Detailed Description

Domain Exception

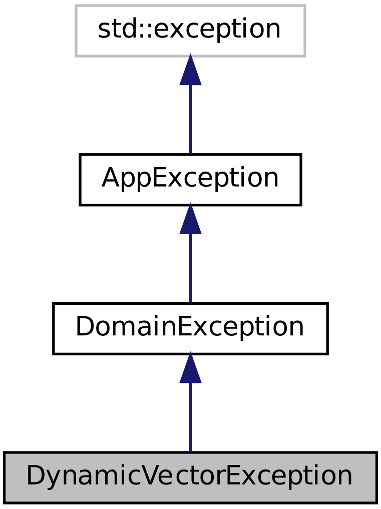
The documentation for this class was generated from the following files:

- include/domain/Exceptions.h
- src/domain/Exceptions.cpp

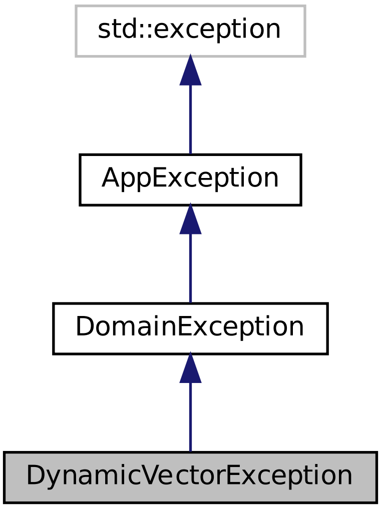
5.8 DynamicVectorException Class Reference

```
#include <Exceptions.h>
```

Inheritance diagram for DynamicVectorException:



Collaboration diagram for DynamicVectorException:



Public Member Functions

- **DynamicVectorException** (const std::string &message)

5.8.1 Detailed Description

DynamicVector exception

The documentation for this class was generated from the following file:

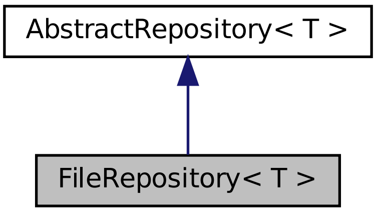
- include/domain/Exceptions.h

5.9 FileRepository< T > Class Template Reference

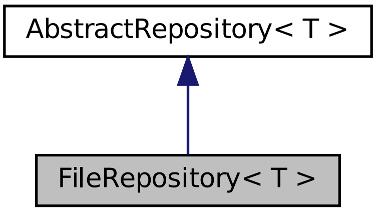
this is the model for a general purpose repository

```
#include <FileRepository.h>
```

Inheritance diagram for FileRepository< T >:



Collaboration diagram for FileRepository< T >:



Public Member Functions

- FileRepository (const std::string &file_path)

  the default constructor of the repository Initializes a repository which holds an empty vector of type T
- ∼FileRepository ()

  the default destructor of the repository
- void add (const T &elem) override

  the add method of the repository. Adds an element to the repository
- void update (const T &elem) override

  the update method of the repository. Updates an element of the repository
- void remove (const T &elem) override

  removes an element using the id provided by an element with the same id.
- T find (const T &elem) override

  returns the element with the same id as the element that is provided
- std::vector< T > get_all () override

  gets all elements from the repository
- void save ()
- void load ()

5.9.1 Detailed Description

template<class T>
class FileRepository< T >

this is the model for a general purpose repository

5.9.2 Constructor & Destructor Documentation

5.9.2.1 FileRepository()

```
template<class T >
FileRepository< T >::FileRepository (
            const std::string & file_path )
```

the default constructor of the repository Initializes a repository which holds an empty vector of type T

Parameters

file_path	this is the path to the file in which to store the data(and from which to load it)

5.9.3 Member Function Documentation

5.9.3.1 add()

```
template<class T >
void FileRepository< T >::add (
            const T & elem )  [override], [virtual]
```

the add method of the repository. Adds an element to the repository

Parameters

elem	the element to be added

Exceptions

RepositoryException	if its id is already taken

Implements AbstractRepository< T >.

5.9.3.2 find()

```
template<class T >
T FileRepository< T >::find (
             const T & elem ) [override], [virtual]
```

returns the element with the same id as the element that is provided

Parameters

elem	the element used to search the id

Returns

the element from the repository with that id

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.9.3.3 get_all()

```
template<class T >
vector< T > FileRepository< T >::get_all [override], [virtual]
```

gets all elements from the repository

Returns

all elements currently held in the repository

Implements AbstractRepository< T >.

5.9.3.4 load() `template<class T >`

`void FileRepository< T >::load`

This function loads the data from a file

5.9.3.5 remove() `template<class T >`

```
void FileRepository< T >::remove (
            const T & elem ) [override], [virtual]
```

removes an element using the id provided by an element with the same id.

Parameters

elem	the element to be removed

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.9.3.6 save() `template<class T >`

`void FileRepository< T >::save`

This function saves the data to a file

5.9.3.7 update() `template<class T >`

```
void FileRepository< T >::update (
            const T & elem ) [override], [virtual]
```

the update method of the repository. Updates an element of the repository

Parameters

elem	an element with which to update. It should have the same id as the one to be updated

Exceptions

RepositoryException	if id doesn't exist

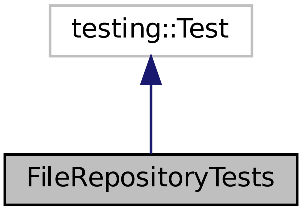
Implements AbstractRepository< T >.

The documentation for this class was generated from the following files:

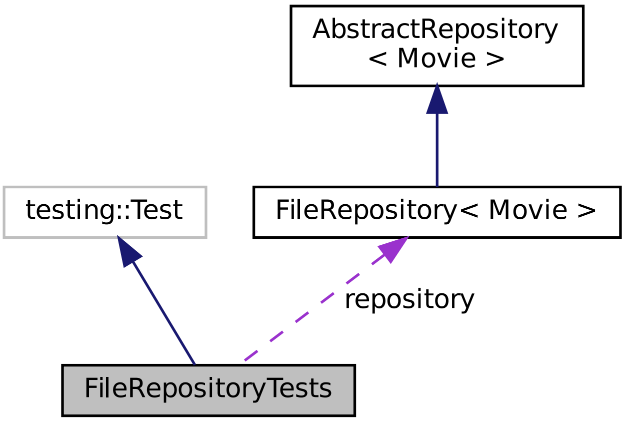
- include/repository/FileRepository.h
- src/repository/FileRepository.cpp

5.10 FileRepositoryTests Class Reference

Inheritance diagram for FileRepositoryTests:



Collaboration diagram for FileRepositoryTests:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

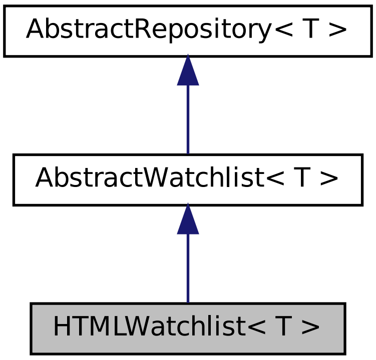
- FileRepository< Movie > **repository**

The documentation for this class was generated from the following file:

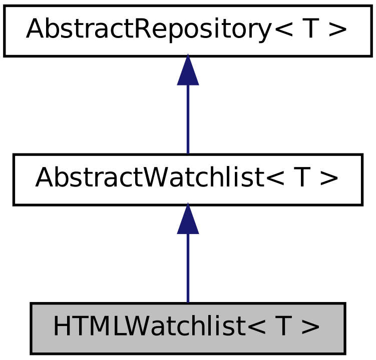
- tests/repository/FileRepositoryTests.cpp

5.11 HTMLWatchlist< T > Class Template Reference

Inheritance diagram for HTMLWatchlist< T >:



Collaboration diagram for HTMLWatchlist< T >:



Public Member Functions

- HTMLWatchlist (const std::string &file_path)

  the default constructor of the repository Initializes a repository which holds an empty vector of type T
- ~HTMLWatchlist ()

  the default destructor of the repository
- void add (const T &elem) override

  the add method of the repository. Adds an element to the repository
- void update (const T &elem) override

the update method of the repository. Updates an element of the repository

- void remove (const T &elem) override

  removes an element using the id provided by an element with the same id.
- T find (const T &elem) override

  returns the element with the same id as the element that is provided
- std::vector< T > get_all () override

  gets all elements from the repository
- void save () const override
- void display () const override

5.11.1 Constructor & Destructor Documentation

5.11.1.1 HTMLWatchlist()

```
template<class T >
HTMLWatchlist< T >::HTMLWatchlist (
            const std::string & file_path )
```

the default constructor of the repository Initializes a repository which holds an empty vector of type T

Parameters

file_path	this is the path to the file in which to store the data

5.11.2 Member Function Documentation

5.11.2.1 add()

```
template<class T >
void HTMLWatchlist< T >::add (
            const T & elem )  [override], [virtual]
```

the add method of the repository. Adds an element to the repository

Parameters

elem	the element to be added

Exceptions

RepositoryException	if its id is already taken

Implements AbstractRepository< T >.

5.11.2.2 display() `template<class T >`

```
void HTMLWatchlist< T >::display  [override], [virtual]
```

This file shows the file

Implements AbstractWatchlist< T >.

5.11.2.3 find() `template<class T >`

```
T HTMLWatchlist< T >::find (
             const T & elem )  [override], [virtual]
```

returns the element with the same id as the element that is provided

Parameters

elem	the element used to search the id

Returns

the element from the repository with that id

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.11.2.4 get_all() `template<class T >`

```
vector< T > HTMLWatchlist< T >::get_all  [override], [virtual]
```

gets all elements from the repository

Returns

all elements currently held in the repository

Implements AbstractRepository< T >.

5.11.2.5 remove() `template<class T >`

```
void HTMLWatchlist< T >::remove (
             const T & elem )  [override], [virtual]
```

removes an element using the id provided by an element with the same id.

Parameters

elem	the element to be removed

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.11.2.6 save()

```
template<class T >
void HTMLWatchlist< T >::save  [override], [virtual]
```

This function saves the data to a file

Implements AbstractWatchlist< T >.

5.11.2.7 update()

```
template<class T >
void HTMLWatchlist< T >::update (
            const T & elem )  [override], [virtual]
```

the update method of the repository. Updates an element of the repository

Parameters

elem	an element with which to update. It should have the same id as the one to be updated

Exceptions

RepositoryException	if id doesn't exist

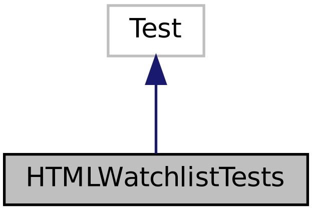
Implements AbstractRepository< T >.

The documentation for this class was generated from the following files:

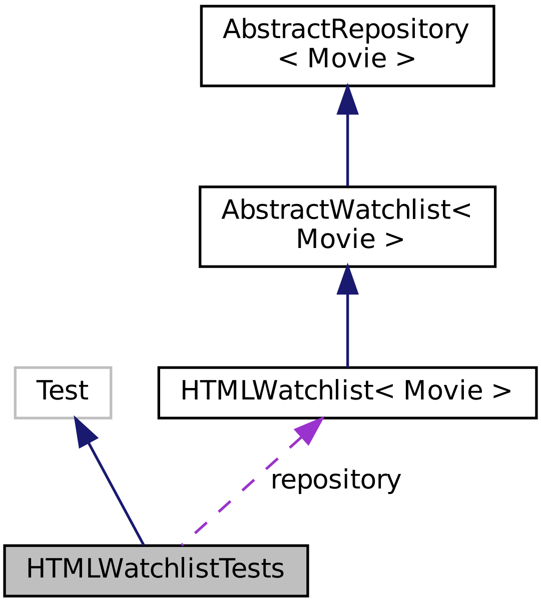
- include/repository/HTMLWatchlist.h
- src/repository/HTMLWatchlist.cpp

5.12 HTMLWatchlistTests Class Reference

Inheritance diagram for HTMLWatchlistTests:



Collaboration diagram for HTMLWatchlistTests:



Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

- HTMLWatchlist< Movie > **repository**

The documentation for this class was generated from the following file:

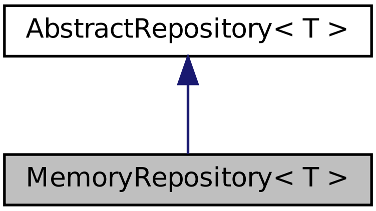
- tests/repository/HTMLWatchlistTests.cpp

5.13 MemoryRepository< T > Class Template Reference

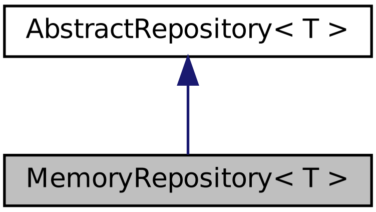
this is the model for a general purpose repository

```
#include <MemoryRepository.h>
```

Inheritance diagram for MemoryRepository< T >:



Collaboration diagram for MemoryRepository< T >:



Public Member Functions

- MemoryRepository ()

    the default constructor of the repository Initializes a repository which holds an empty vector of type T
- ∼MemoryRepository ()

    the default destructor of the repository
- void add (const T &elem) override

    the add method of the repository. Adds an element to the repository
- void update (const T &elem) override

    the update method of the repository. Updates an element of the repository
- void remove (const T &elem) override

    removes an element using the id provided by an element with the same id.
- T find (const T &elem) override

    returns the element with the same id as the element that is provided
- std::vector< T > get_all () override

    gets all elements from the repository

5.13.1 Detailed Description

template<class T>
class MemoryRepository< T >

this is the model for a general purpose repository

5.13.2 Member Function Documentation

5.13.2.1 add()

```
template<class T >
void MemoryRepository< T >::add (
            const T & elem )  [override], [virtual]
```

the add method of the repository. Adds an element to the repository

Parameters

elem	the element to be added

Exceptions

RepositoryException	if its id is already taken

Implements AbstractRepository< T >.

5.13.2.2 find() `template<class T >`

```
T MemoryRepository< T >::find (
             const T & elem )  [override], [virtual]
```

returns the element with the same id as the element that is provided

Parameters

elem	the element used to search the id

Returns

the element from the repository with that id

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.13.2.3 get_all() `template<class T >`

```
vector< T > MemoryRepository< T >::get_all  [override], [virtual]
```

gets all elements from the repository

Returns

all elements currently held in the repository

Implements AbstractRepository< T >.

5.13.2.4 remove() `template<class T >`

```
void MemoryRepository< T >::remove (
             const T & elem )  [override], [virtual]
```

removes an element using the id provided by an element with the same id.

Parameters

elem	the element to be removed

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.13.2.5 update() `template<class T >`

```
void MemoryRepository< T >::update (
             const T & elem )  [override], [virtual]
```

the update method of the repository. Updates an element of the repository

Parameters

elem	an element with which to update. It should have the same id as the one to be updated

Exceptions

RepositoryException	if id doesn't exist

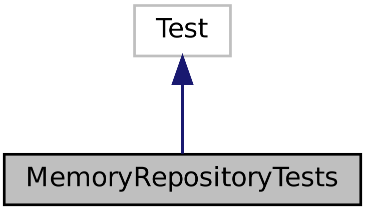
Implements AbstractRepository< T >.

The documentation for this class was generated from the following files:

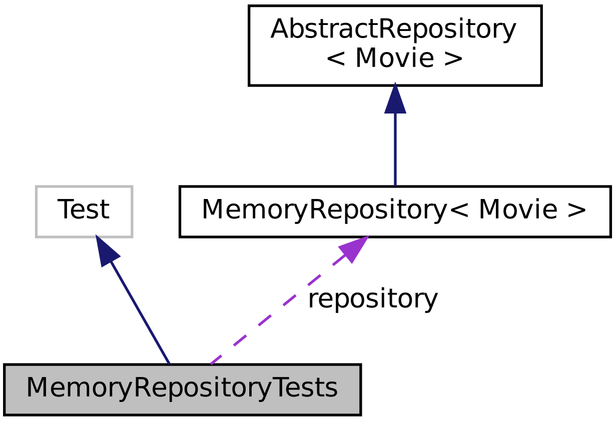
- include/repository/MemoryRepository.h
- src/repository/MemoryRepository.cpp

5.14 MemoryRepositoryTests Class Reference

Inheritance diagram for MemoryRepositoryTests:



Collaboration diagram for MemoryRepositoryTests:



Protected Member Functions

- void **SetUp** () override

Protected Attributes

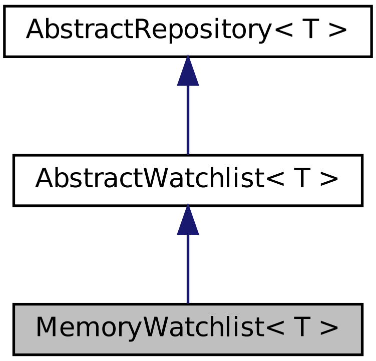
- MemoryRepository< Movie > **repository**

The documentation for this class was generated from the following file:

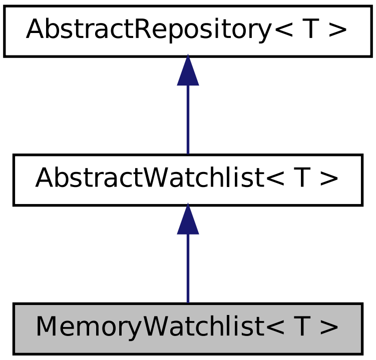
- tests/repository/MemoryRepositoryTests.cpp

5.15 MemoryWatchlist< T > Class Template Reference

Inheritance diagram for MemoryWatchlist< T >:



Collaboration diagram for MemoryWatchlist< T >:



Public Member Functions

- MemoryWatchlist ()

  the default constructor of the repository Initializes a repository which holds an empty vector of type T
- ∼MemoryWatchlist ()

  the default destructor of the repository
- void add (const T &elem) override

  the add method of the repository. Adds an element to the repository
- void update (const T &elem) override

  the update method of the repository. Updates an element of the repository
- void remove (const T &elem) override

  removes an element using the id provided by an element with the same id.
- T find (const T &elem) override

  returns the element with the same id as the element that is provided
- std::vector< T > get_all () override

  gets all elements from the repository
- void save () const override
- void display () const override

5.15.1 Member Function Documentation

5.15.1.1 add()

```
template<class T >
void MemoryWatchlist< T >::add (
            const T & elem )  [override], [virtual]
```

the add method of the repository. Adds an element to the repository

Parameters

elem	the element to be added

Exceptions

RepositoryException	if its id is already taken

Implements AbstractRepository< T >.

5.15.1.2 display()

```
template<class T >
void MemoryWatchlist< T >::display  [override], [virtual]
```

This file shows the file

Implements AbstractWatchlist< T >.

5.15.1.3 find()

```
template<class T >
T MemoryWatchlist< T >::find (
             const T & elem )  [override], [virtual]
```

returns the element with the same id as the element that is provided

Parameters

elem	the element used to search the id

Returns

the element from the repository with that id

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.15.1.4 get_all()

```
template<class T >
vector< T > MemoryWatchlist< T >::get_all  [override], [virtual]
```

gets all elements from the repository

Returns

all elements currently held in the repository

Implements AbstractRepository< T >.

5.15.1.5 remove()

```
template<class T >
void MemoryWatchlist< T >::remove (
             const T & elem )  [override], [virtual]
```

removes an element using the id provided by an element with the same id.

Parameters

elem	the element to be removed

Exceptions

RepositoryException	if id doesn't exist

Implements AbstractRepository< T >.

5.15.1.6 save()

```
template<class T >
void MemoryWatchlist< T >::save  [override], [virtual]
```

This function saves the data to a file

Implements AbstractWatchlist< T >.

5.15.1.7 update()

```
template<class T >
void MemoryWatchlist< T >::update (
             const T & elem )  [override], [virtual]
```

the update method of the repository. Updates an element of the repository

Parameters

elem	an element with which to update. It should have the same id as the one to be updated

Exceptions

RepositoryException	if id doesn't exist

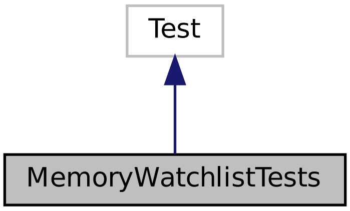
Implements AbstractRepository< T >.

The documentation for this class was generated from the following files:

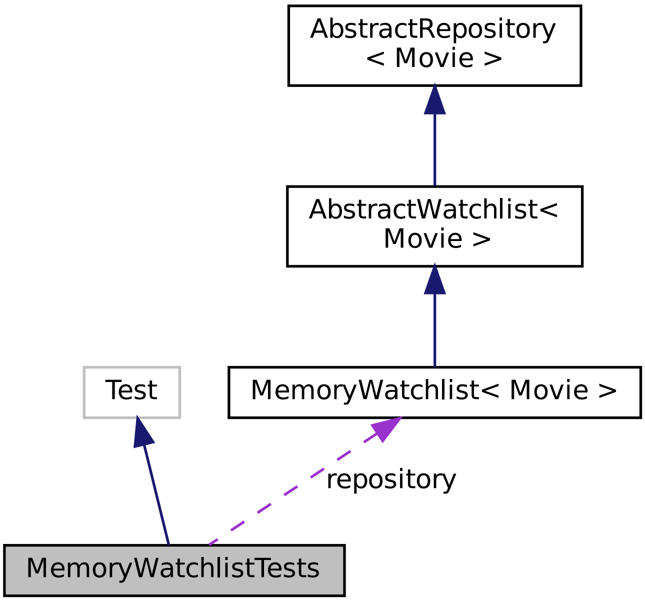
- include/repository/MemoryWatchlist.h
- src/repository/MemoryWatchlist.cpp

5.16 MemoryWatchlistTests Class Reference

Inheritance diagram for MemoryWatchlistTests:



Collaboration diagram for MemoryWatchlistTests:



Protected Member Functions

- void **SetUp** () override

Protected Attributes

- MemoryWatchlist< Movie > **repository**

The documentation for this class was generated from the following file:

- tests/repository/MemoryWatchlistTests.cpp

5.17 Movie Class Reference

```
#include <Movie.h>
```

Public Member Functions

- Movie ()

  default constructor for an Movie instance
- Movie (const Movie &other)
- Movie (const std::string &title, const std::string &genre, const int &year, const int &likes, const std::string &trailer_link)
- std::string get_title () const

  gets the title of the movie
- std::string get_genre () const

  gets the genre of the movie
- int get_year () const

  gets the year of the movie
- int get_likes () const

  gets the number of likes of the movie
- std::string get_trailer_link () const

  gets the link to the trailer
- void set_title (const std::string &title)

  sets the title of the movie
- void set_genre (const std::string &genre)

  sets the genre of the movie
- void set_year (int year)

  sets the year of the movie
- void set_likes (int likes)

  sets the number of likes of the movie
- void set_trailer_link (const std::string &trailer_link)

  set the trailer link
- std::string get_id () const

  this function returns a string that will correspond to the id of the movie
- bool operator== (const Movie &other) const

  equality operator between movies
- bool operator!= (const Movie &other) const

  not equal operator between movies

Friends

- std::istream & operator>> (std::istream &in, Movie &c)

  istream insertion operator
- std::ostream & operator<< (std::ostream &out, Movie c)

  ostream extraction operator

5.17.1 Detailed Description

The Movie class model

5.17.2 Constructor & Destructor Documentation

5.17.2.1 Movie() [1/2]

```
Movie::Movie (
        const Movie & other )
```

copy constructor for an Movie instance

Parameters

other	the other movie instance who's attributes are copied.

5.17.2.2 Movie() [2/2]

```
Movie::Movie (
        const std::string & title,
        const std::string & genre,
        const int & year,
        const int & likes,
        const std::string & trailer_link )
```

Constructor with full params.

Parameters

title	the movie title
genre	the movie genre
year	the year in which the movie was released
likes	the number of likes the movie has
trailer_link	a link to the trailer of the movie

5.17.3 Member Function Documentation

5.17.3.1 get_genre() `std::string Movie::get_genre ( ) const`

gets the genre of the movie

Returns

the genre of the movie

5.17.3.2 get_id() `std::string Movie::get_id ( ) const`

this function returns a string that will correspond to the id of the movie

Returns

the id of the movie

5.17.3.3 get_likes() `int Movie::get_likes ( ) const`

gets the number of likes of the movie

Returns

the number of likes that the movie has

5.17.3.4 get_title() `std::string Movie::get_title ( ) const`

gets the title of the movie

Returns

the title of the movies

5.17.3.5 get_trailer_link() `std::string Movie::get_trailer_link ( ) const`

gets the link to the trailer

Returns

the link to the movie

5.17.3.6 get_year() `int Movie::get_year ( ) const`

gets the year of the movie

Returns

the year in which the movie was released

5.17.3.7 operator"!=() `bool Movie::operator!= (`

```
        const Movie & other ) const
```

not equal operator between movies

Parameters

other	the other movie

Returns

false if they are the same, true otherwise

5.17.3.8 operator==()

```
bool Movie::operator== (
        const Movie & other ) const
```

equality operator between movies

Parameters

other	the other movie

Returns

true if they are the same, false otherwise

5.17.3.9 set_genre()

```
void Movie::set_genre (
        const std::string & genre )
```

sets the genre of the movie

Parameters

genre	the new genre of the movie

5.17.3.10 set_likes()

```
void Movie::set_likes (
        int likes )
```

sets the number of likes of the movie

Parameters

likes	the nw likes of the movie

5.17.3.11 set_title()

```
void Movie::set_title (
        const std::string & title )
```

sets the title of the movie

Parameters

title	the new title of the movie

5.17.3.12 set_trailer_link()

```
void Movie::set_trailer_link (
        const std::string & trailer_link )
```

set the trailer link

Parameters

trailer_link	the new trailer link of the movie

5.17.3.13 set_year()

```
void Movie::set_year (
        int year )
```

sets the year of the movie

Parameters

year	the new year of the movie

5.17.4 Friends And Related Function Documentation

5.17.4.1 operator<<

```
std::ostream& operator<< (
        std::ostream & out,
        Movie c ) [friend]
```

ostream extraction operator

Parameters

out	
c	

Returns

5.17.4.2 operator>> `std::istream& operator>> ( std::istream & in, Movie & c ) [friend]`

istream insertion operator

Parameters

in	the input stream
c	the movie

Returns

the input stream

The documentation for this class was generated from the following files:

- include/domain/Movie.h
- src/domain/Movie.cpp

5.18 MovieController Class Reference

the model of the MovieController class

`#include <MovieController.h>`

Public Member Functions

- MovieController (AbstractRepository< Movie > &repository, AbstractWatchlist< Movie > &watchlist)

  The default constructor of the movie controller.
- ∼MovieController ()

  The destructor of the movie controller.
- MovieController & operator= (const MovieController &other)

  the assignment operator
- void add_movie_admin (const std::string &title, const std::string &genre, const int &year, const int &likes, const std::string &trailer_link)

  adds a movie to the movie repository
- void update_movie_admin (const std::string &title, const std::string &genre, const int &year, const int &likes, const std::string &trailer_link)

  updates a movie to the movie repository
- void remove_movie_admin (const std::string &title, const int &year)

  this function removes a movie from the movie repository
- Movie get_movie_user (const std::string &title, const int &year)

this function finds a movie in the repository
- std::vector< Movie > get_all_movies_admin ()

  this function returns all the movies from the repository
- std::vector< Movie > get_movies_user (const std::string genre)

  this function gets a vector with all the movies that have a given genre. If the genre is empty, all genres are considered.
- void add_movie_user (const Movie &movie)

  this function adds a movie to the user watch list
- void remove_movie_user (const Movie &movie)

  this function removes a movie from the user watch list
- std::vector< Movie > get_watchlist_user ()

  this function gets a list of the movies in the user's watchlist
- void like_movie_user (const Movie &movie)

  gives a like to a movie
- std::string my_tolower (const std::string &s) const

  converts a string to lower characters
- void display_watchlist () const

  this function displays the current watchlist, in the required format.

5.18.1 Detailed Description

the model of the MovieController class

5.18.2 Member Function Documentation

5.18.2.1 add_movie_admin()

```
void MovieController::add_movie_admin (
        const std::string & title,
        const std::string & genre,
        const int & year,
        const int & likes,
        const std::string & trailer_link )
```

adds a movie to the movie repository

Parameters

title	the title of the movie to be added
genre	the genre of the movie to be added
year	the year of the movie to be added
likes	the number of likes of the movie to be added
trailer_link	the trailer of the movie to be added

5.18.2.2 add_movie_user()

```
void MovieController::add_movie_user (
        const Movie & movie )
```

this function adds a movie to the user watch list

Parameters

movie	the movie to be added

5.18.2.3 get_all_movies_admin()

```
vector< Movie > MovieController::get_all_movies_admin ( )
```

this function returns all the movies from the repository

Returns

a list of the movies from the repository

5.18.2.4 get_movie_user()

```
Movie MovieController::get_movie_user (
        const std::string & title,
        const int & year )
```

this function finds a movie in the repository

Parameters

title	the title of the movie
year	the year of the movie

5.18.2.5 get_movies_user()

```
vector< Movie > MovieController::get_movies_user (
        const std::string genre )
```

this function gets a vector with all the movies that have a given genre. If the genre is empty, all genres are considered.

Parameters

genre	the genre of the movies

Returns

a vector of the movies

5.18.2.6 like_movie_user()

```
void MovieController::like_movie_user (
        const Movie & movie )
```

gives a like to a movie

Parameters

movie	the movie to be liked

5.18.2.7 my_tolower()

```
std::string MovieController::my_tolower (
        const std::string & s ) const
```

converts a string to lower characters

Parameters

s	the string

Returns

the string containing only lowered characters

5.18.2.8 operator=()

```
MovieController & MovieController::operator= (
        const MovieController & other )
```

the assignment operator

Parameters

other	the other controller

Returns

the copy

5.18.2.9 remove_movie_admin()

```
void MovieController::remove_movie_admin (
        const std::string & title,
        const int & year )
```

this function removes a movie from the movie repository

Parameters

title	the title of the movie
year	the year of the movie

5.18.2.10 remove_movie_user()

```
void MovieController::remove_movie_user (
            const Movie & movie )
```

this function removes a movie from the user watch list

Parameters

movie	the movie to be removed

5.18.2.11 update_movie_admin()

```
void MovieController::update_movie_admin (
            const std::string & title,
            const std::string & genre,
            const int & year,
            const int & likes,
            const std::string & trailer_link )
```

updates a movie to the movie repository

Parameters

title	the title of the movie to be added
genre	the genre of the movie to be added
year	the year of the movie to be added
likes	the number of likes of the movie to be added
trailer_link	the trailer of the movie to be added

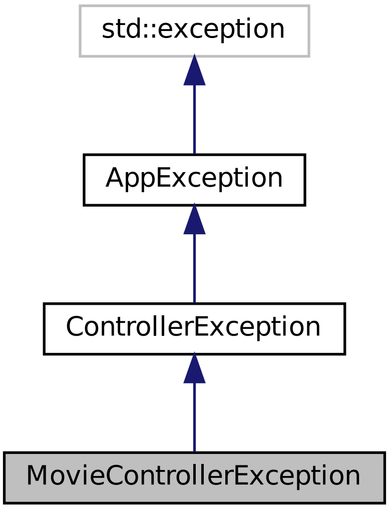
The documentation for this class was generated from the following files:

- include/controller/MovieController.h
- src/controller/MovieController.cpp

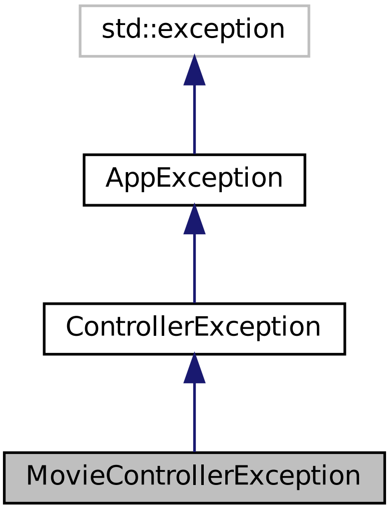
5.19 MovieControllerException Class Reference

```
#include <Exceptions.h>
```

Inheritance diagram for MovieControllerException:



Collaboration diagram for MovieControllerException:



Public Member Functions

- **MovieControllerException** (const std::string &message)

5.19.1 Detailed Description

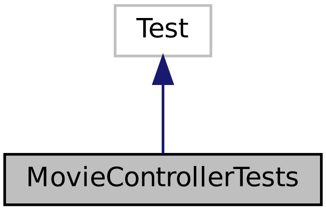
Controller Exception

The documentation for this class was generated from the following files:

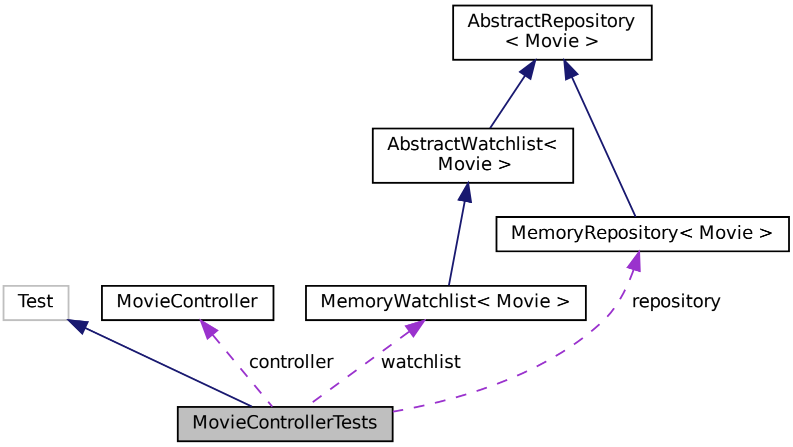
- include/domain/Exceptions.h
- src/domain/Exceptions.cpp

5.20 MovieControllerTests Class Reference

Inheritance diagram for MovieControllerTests:



Collaboration diagram for MovieControllerTests:



Protected Member Functions

- void **SetUp** () override

Protected Attributes

- MovieController **controller**
- vector< Movie > **movies**
- MemoryWatchlist< Movie > **watchlist**
- MemoryRepository< Movie > **repository**

The documentation for this class was generated from the following file:

- tests/controller/MovieControllerTests.cpp

5.21 MovieValidator Class Reference

the movie validator

```
#include <MovieValidator.h>
```

Static Public Member Functions

- static void validate_movie (const Movie &movie)

    static method that checks if a given movie is valid

5.21.1 Detailed Description

the movie validator

5.21.2 Member Function Documentation

5.21.2.1 validate_movie()

```
void MovieValidator::validate_movie (
            const Movie & movie )  [static]
```

static method that checks if a given movie is valid

Parameters

movie	the movie

Exceptions

DomainException	if movie invalid

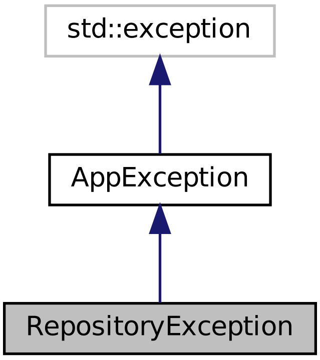
The documentation for this class was generated from the following files:

- include/domain/MovieValidator.h
- src/domain/MovieValidator.cpp

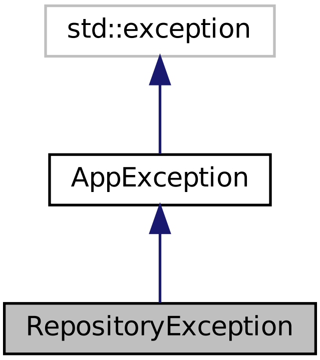
5.22 RepositoryException Class Reference

```
#include <Exceptions.h>
```

Inheritance diagram for RepositoryException:



Collaboration diagram for RepositoryException:



Public Member Functions

- **RepositoryException** (const std::string &message)

5.22.1 Detailed Description

Repository Exception

The documentation for this class was generated from the following files:

- include/domain/Exceptions.h
- src/domain/Exceptions.cpp

5.23 UI Class Reference

this class contains the model for an ui class

```
#include <UI.h>
```

Public Member Functions

- UI (MovieController &controller)

  the constructor of an UI instance
- void run ()

5.23.1 Detailed Description

this class contains the model for an ui class

5.23.2 Constructor & Destructor Documentation

5.23.2.1 UI()

```
UI::UI (
        MovieController & controller )
```

the constructor of an UI instance

Parameters

controller	the movie controller to be used

5.23.3 Member Function Documentation

5.23.3.1 run()

```
void UI::run ( )
```

runs the ui

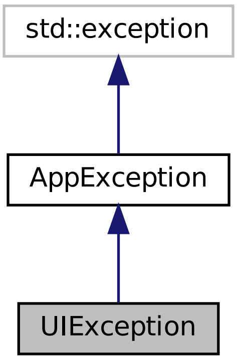
The documentation for this class was generated from the following files:

- include/ui/UI.h
- src/ui/UI.cpp

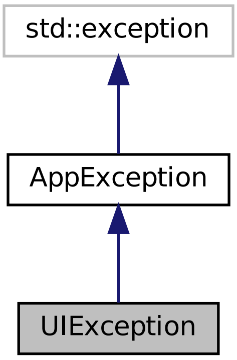
5.24 UIException Class Reference

```
#include <Exceptions.h>
```

Inheritance diagram for UIException:



Collaboration diagram for UIException:



Public Member Functions

- **UIException** (const std::string &message)

5.24.1 Detailed Description

UI Exception

The documentation for this class was generated from the following files:

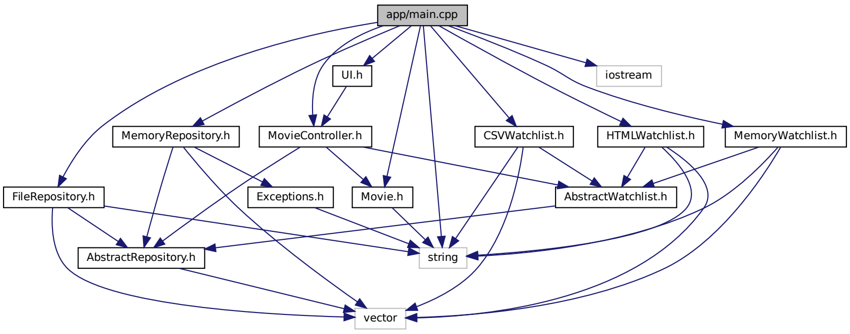
- include/domain/Exceptions.h
- src/domain/Exceptions.cpp

6 File Documentation

6.1 app/main.cpp File Reference

```
#include "Movie.h"
#include "MemoryRepository.h"
#include "MemoryWatchlist.h"
#include "MovieController.h"
#include "FileRepository.h"
#include "CSVWatchlist.h"
#include "HTMLWatchlist.h"
#include "UI.h"
#include <iostream>
#include <string>
```

Include dependency graph for main.cpp:



Functions

- int **main** ()

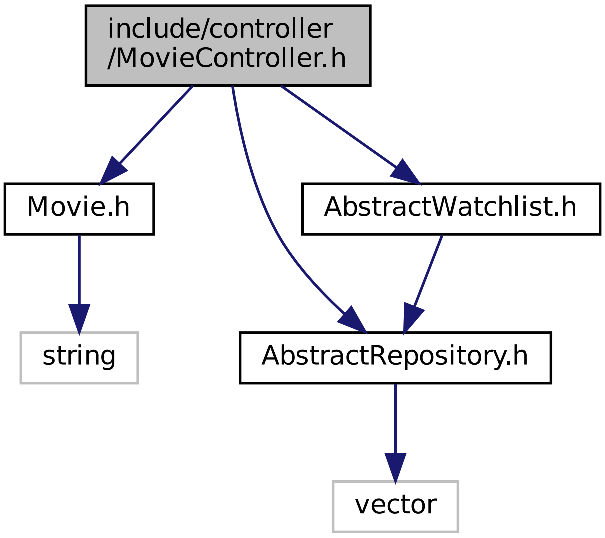
6.1.1 Detailed Description

the main file of the application

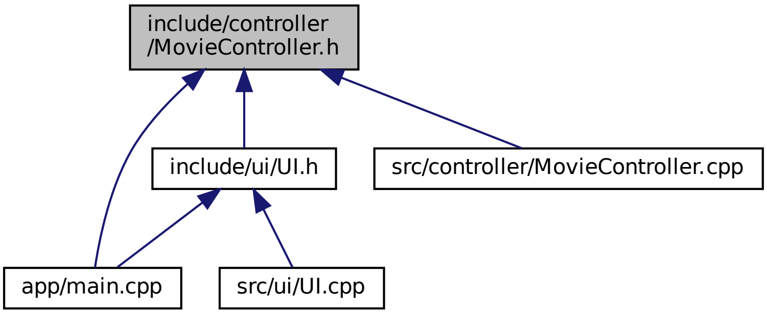
6.2 include/controller/MovieController.h File Reference

```
#include "Movie.h"
#include "AbstractRepository.h"
#include "AbstractWatchlist.h"
```

Include dependency graph for MovieController.h:



This graph shows which files directly or indirectly include this file:



Classes

- class MovieController

  the model of the MovieController class

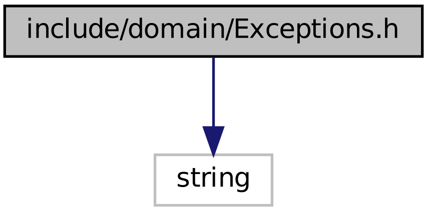
6.2.1 Detailed Description

this file contains the model for a movie controller

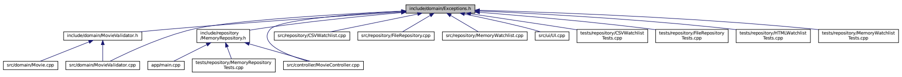
6.3 include/domain/Exceptions.h File Reference

```
#include <string>
```

Include dependency graph for Exceptions.h:



This graph shows which files directly or indirectly include this file:


include/domain/Exceptions.h
include/domain/MovieValidator.h
include/repository
/MemoryRepository.h
src/repository/CSVWatchlist.cpp
src/repository/FileRepository.cpp
src/repository/MemoryWatchlist.cpp
src/ui/UI.cpp
tests/repository/CSVWatchlist
Tests.cpp
tests/repository/FileRepository
Tests.cpp
tests/repository/HTMLWatchlist
Tests.cpp
tests/repository/MemoryWatchlist
Tests.cpp
src/domain/Movie.cpp
src/domain/MovieValidator.cpp
app/main.cpp
tests/repository/MemoryRepository
Tests.cpp
src/controller/MovieController.cpp

Classes

- class AppException
- class DomainException
- class DynamicVectorException
- class RepositoryException
- class ControllerException
- class MovieControllerException
- class UIException

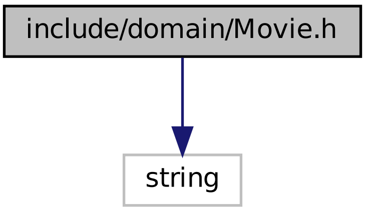
6.3.1 Detailed Description

this file contains the models for all exceptions used in app.

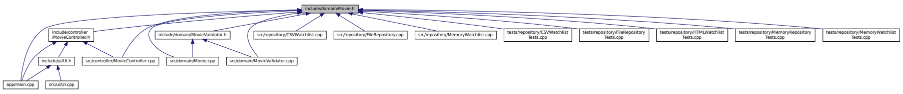
6.4 include/domain/Movie.h File Reference

```
#include <string>
```

Include dependency graph for Movie.h:



This graph shows which files directly or indirectly include this file:


include/domain/Movie.h
include/controller
/MovieController.h
include/domain/MovieValidator.h
src/repository/CSVWatchlist.cpp
src/repository/FileRepository.cpp
src/repository/MemoryWatchlist.cpp
tests/repository/CSVWatchlist
Tests.cpp
tests/repository/FileRepository
Tests.cpp
tests/repository/HTMLWatchlist
Tests.cpp
tests/repository/MemoryRepository
Tests.cpp
tests/repository/MemoryWatchlist
Tests.cpp
include/ui/UI.h
src/controller/MovieController.cpp
src/domain/Movie.cpp
src/domain/MovieValidator.cpp
app/main.cpp
src/ui/UI.cpp

Classes

- class Movie

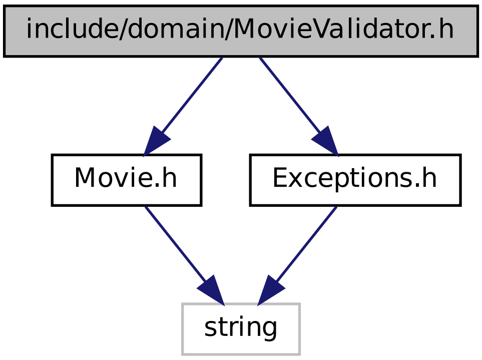
6.4.1 Detailed Description

this file contains the model for the Movie class

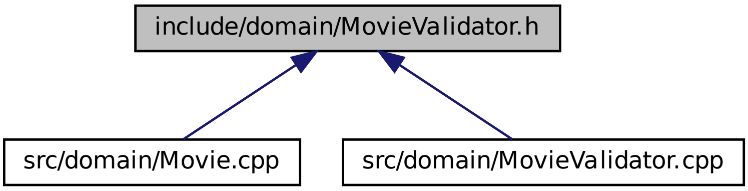
6.5 include/domain/MovieValidator.h File Reference

```
#include "Movie.h"
#include "Exceptions.h"
```

Include dependency graph for MovieValidator.h:



This graph shows which files directly or indirectly include this file:


include/domain/MovieValidator.h
src/domain/Movie.cpp
src/domain/MovieValidator.cpp

Classes

- class MovieValidator

  the movie validator

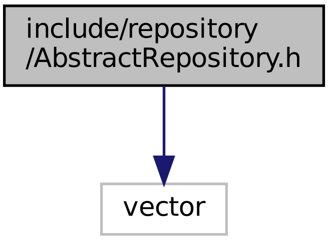
6.5.1 Detailed Description

this file contains the model for the movie validator

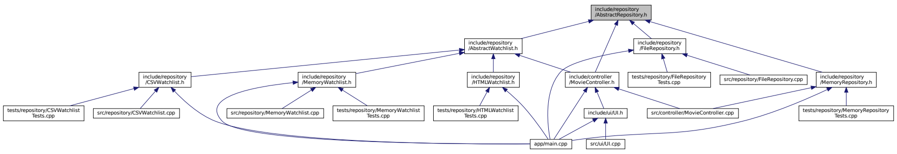
6.6 include/repository/AbstractRepository.h File Reference

```
#include <vector>
```

Include dependency graph for AbstractRepository.h:



This graph shows which files directly or indirectly include this file:



Classes

- class AbstractRepository< T >

  this is the model for a general purpose repository

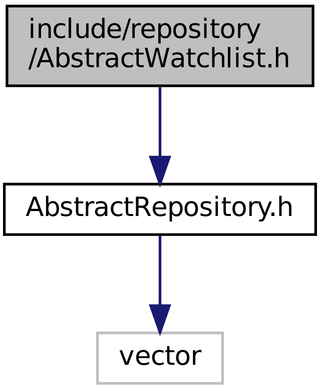
6.6.1 Detailed Description

this file contains the model for a completely abstract repository

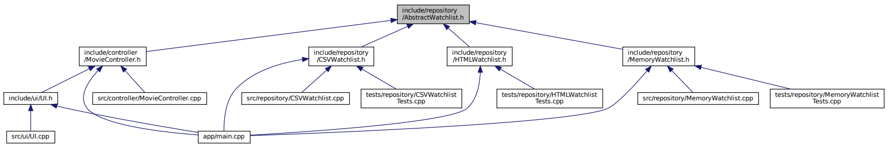
6.7 include/repository/AbstractWatchlist.h File Reference

```
#include "AbstractRepository.h"
```

Include dependency graph for AbstractWatchlist.h:



This graph shows which files directly or indirectly include this file:



Classes

- class AbstractWatchlist< T >

  this struct contains the model for an AbstractWatchlist

6.7.1 Detailed Description

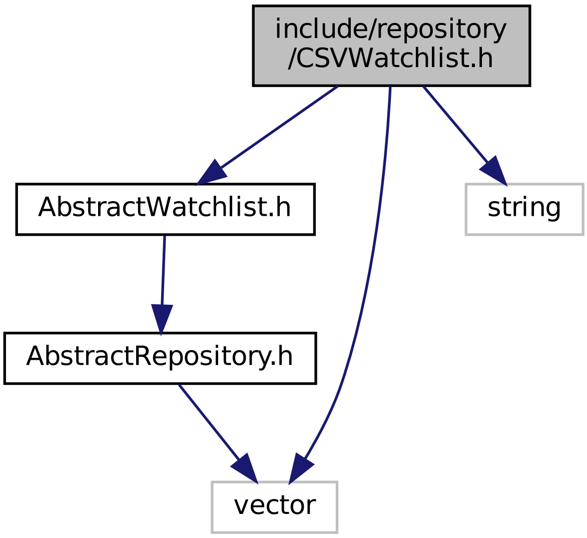
this file contains the model for an abstract watchlist

6.8 include/repository/CSVWatchlist.h File Reference

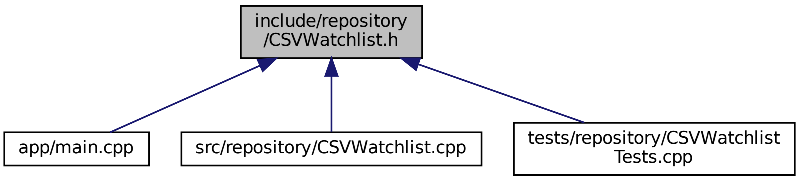
```
#include "AbstractWatchlist.h"
#include <vector>
```

```
#include <string>
```

Include dependency graph for CSVWatchlist.h:



This graph shows which files directly or indirectly include this file:


include/repository
/CSVWatchlist.h
app/main.cpp
src/repository/CSVWatchlist.cpp
tests/repository/CSVWatchlist
Tests.cpp

Classes

- class CSVWatchlist< T >

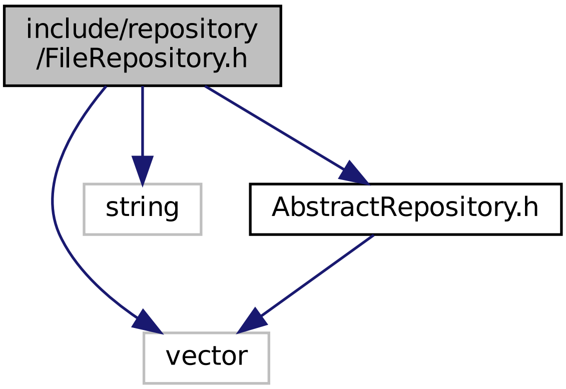
6.8.1 Detailed Description

this is a file that shows the model for a CSVWatchlist

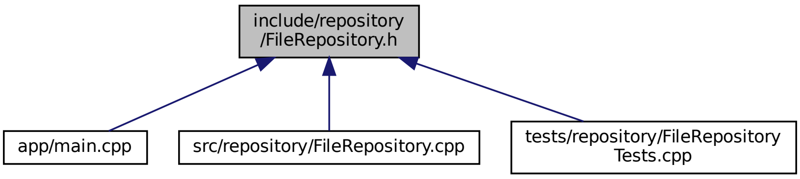
6.9 include/repository/FileRepository.h File Reference

```
#include <vector>
#include <string>
#include "AbstractRepository.h"
```

Include dependency graph for FileRepository.h:



This graph shows which files directly or indirectly include this file:


include/repository
/FileRepository.h
app/main.cpp
src/repository/FileRepository.cpp
tests/repository/FileRepository
Tests.cpp

Classes

- class FileRepository< T >

  this is the model for a general purpose repository

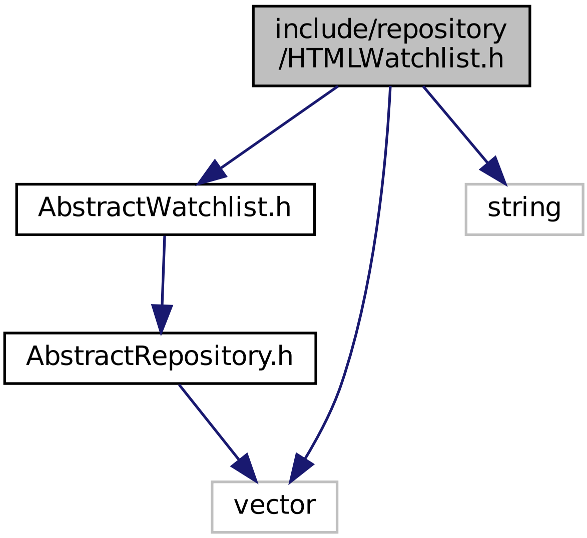
6.9.1 Detailed Description

this file contains the model for a file repository

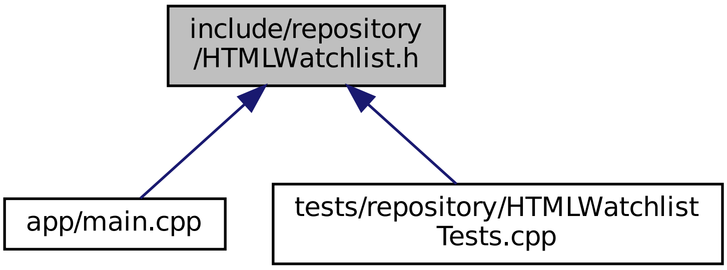
6.10 include/repository/HTMLWatchlist.h File Reference

```
#include "AbstractWatchlist.h"
#include <vector>
#include <string>
```

Include dependency graph for HTMLWatchlist.h:



This graph shows which files directly or indirectly include this file:



Classes

- class HTMLWatchlist< T >

6.10.1 Detailed Description

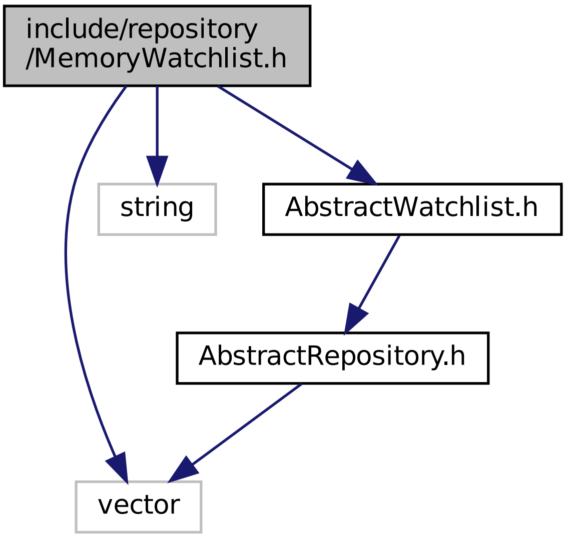
this is a file that shows the model for a CSVWatchlist

this is a file that shows the implementation for a HTMLWatchlist

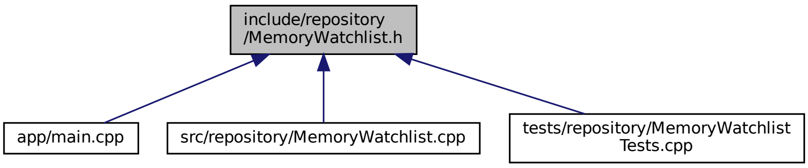
6.11 include/repository/MemoryWatchlist.h File Reference

```
#include <vector>
#include <string>
#include "AbstractWatchlist.h"
```

Include dependency graph for MemoryWatchlist.h:



This graph shows which files directly or indirectly include this file:


include/repository
/MemoryWatchlist.h
app/main.cpp
src/repository/MemoryWatchlist.cpp
tests/repository/MemoryWatchlist
Tests.cpp

Classes

- class MemoryWatchlist< T >

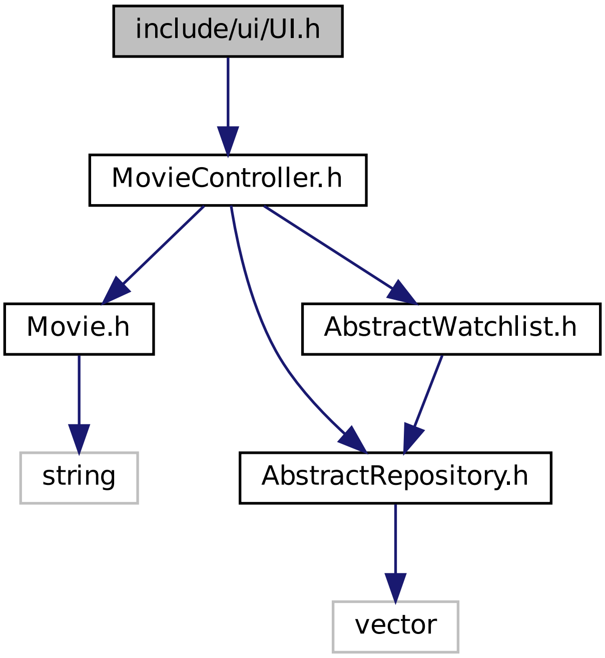
6.11.1 Detailed Description

this is the model for an in-memory watchlist

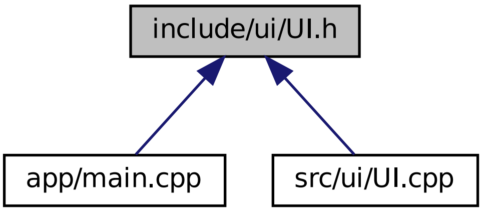
6.12 include/ui/UI.h File Reference

```
#include "MovieController.h"
```

Include dependency graph for UI.h:



This graph shows which files directly or indirectly include this file:



Classes

- class UI

  this class contains the model for an ui class

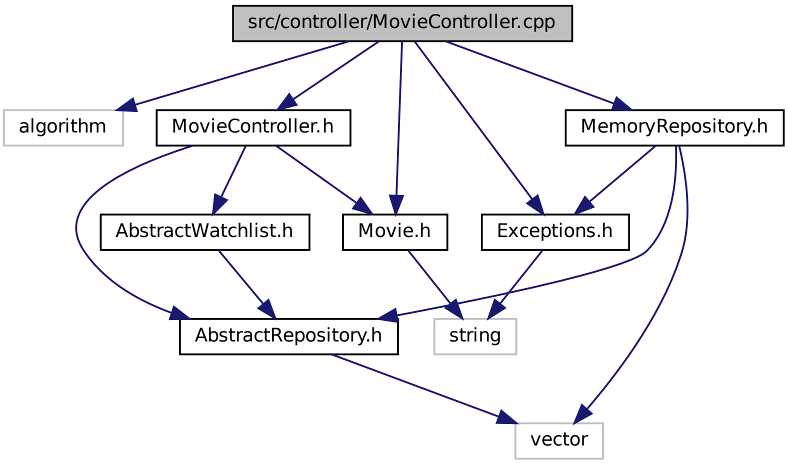
6.12.1 Detailed Description

this file contains the model for an UI class

6.13 src/controller/MovieController.cpp File Reference

```
#include <algorithm>
#include "MovieController.h"
#include "MemoryRepository.h"
#include "Movie.h"
#include "Exceptions.h"
```

Include dependency graph for MovieController.cpp:



6.13.1 Detailed Description

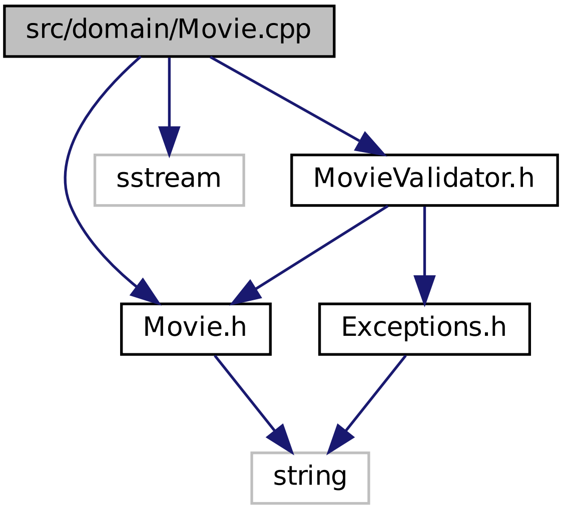
this file contains the implementations for MovieController

6.14 src/domain/Movie.cpp File Reference

```
#include "Movie.h"
#include <sstream>
```

```
#include "MovieValidator.h"
```

Include dependency graph for Movie.cpp:



Functions

- std::istream & operator>> (std::istream &in, Movie &c)
- std::ostream & operator<< (std::ostream &out, Movie c)

6.14.1 Detailed Description

this file contains the implementation for the Movie class

6.14.2 Function Documentation

6.14.2.1 operator<<()

```
std::ostream& operator<< (
    std::ostream & out,
    Movie c )
```

Parameters

out	
c	

Returns

6.14.2.2 operator>>() `std::istream& operator>> ( std::istream & in, Movie & c )`

Parameters

in	the input stream
c	the movie

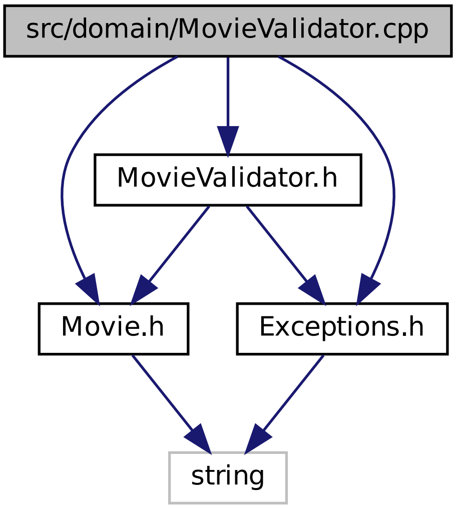
Returns

the input stream

6.15 src/domain/MovieValidator.cpp File Reference

```
#include "Movie.h"
#include "MovieValidator.h"
#include "Exceptions.h"
```

Include dependency graph for MovieValidator.cpp:



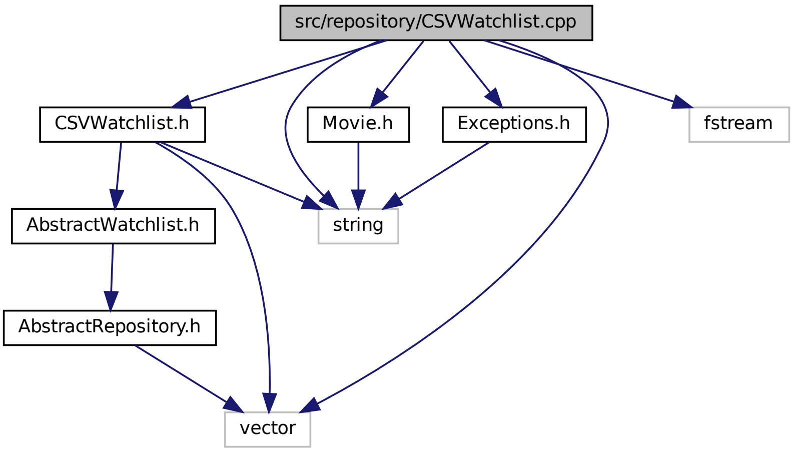
6.15.1 Detailed Description

this file contains the implementation for the movie validator

6.16 src/repository/CSVWatchlist.cpp File Reference

```
#include "CSVWatchlist.h"
#include "Movie.h"
#include "Exceptions.h"
#include <string>
#include <vector>
#include <fstream>
```

Include dependency graph for CSVWatchlist.cpp:



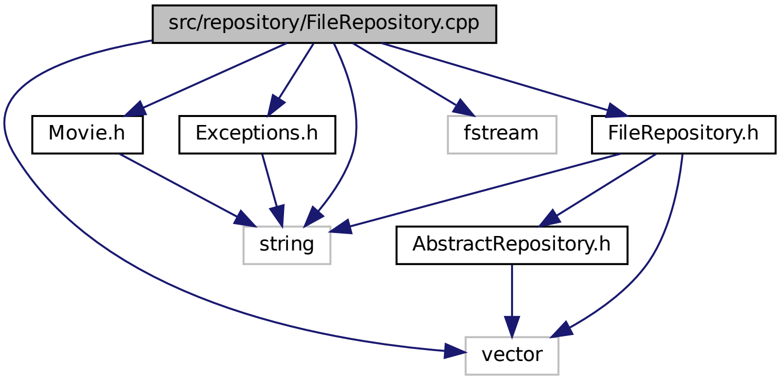
6.16.1 Detailed Description

this file contains the implementation for the CSV watchlist

6.17 src/repository/FileRepository.cpp File Reference

```
#include "FileRepository.h"
#include "Movie.h"
#include <vector>
#include <string>
#include <fstream>
#include "Exceptions.h"
```

Include dependency graph for FileRepository.cpp:



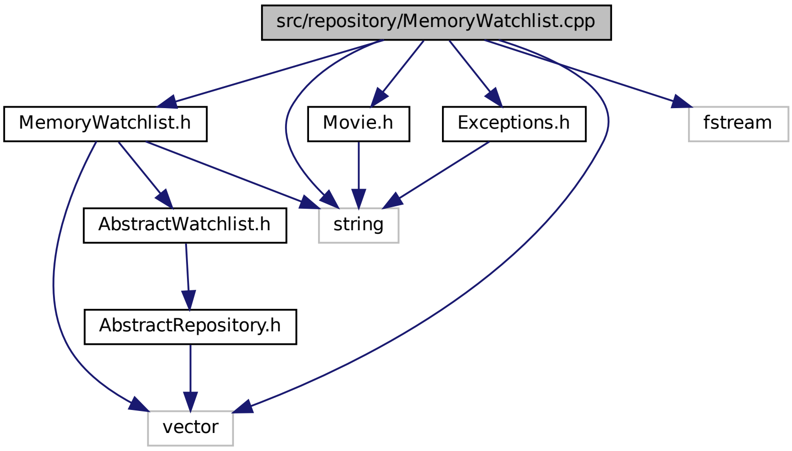
6.17.1 Detailed Description

this file contains the implementation for a file repository

6.18 src/repository/MemoryWatchlist.cpp File Reference

```
#include "MemoryWatchlist.h"
#include "Movie.h"
#include "Exceptions.h"
#include <string>
#include <vector>
#include <fstream>
```

Include dependency graph for MemoryWatchlist.cpp:



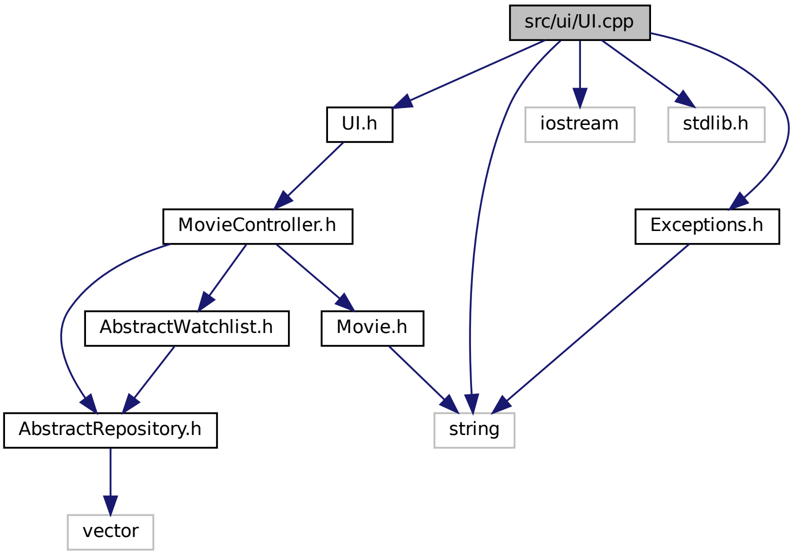
6.18.1 Detailed Description

this is a file containing the implementation for an in-memory watchlist

6.19 src/ui/UI.cpp File Reference

```
#include "UI.h"
#include <string>
#include <iostream>
#include <stdlib.h>
#include "Exceptions.h"
```

Include dependency graph for UI.cpp:



6.19.1 Detailed Description

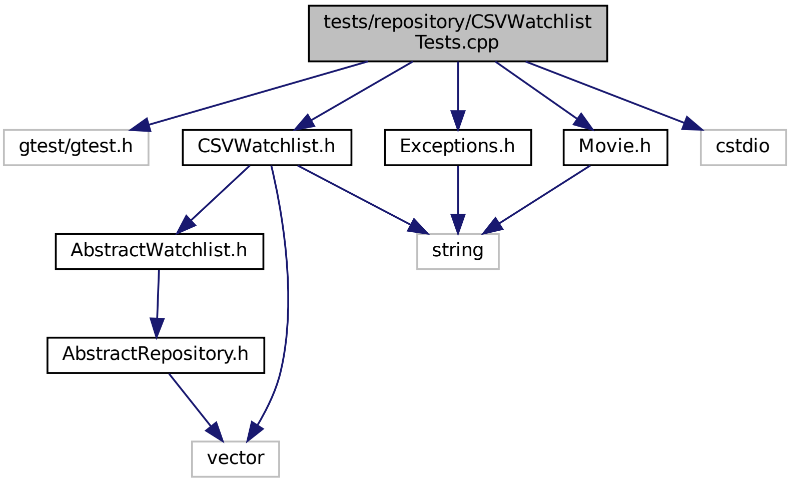
this file contains the UI class implementation

6.20 tests/repository/CSVWatchlistTests.cpp File Reference

```
#include "gtest/gtest.h"
#include "CSVWatchlist.h"
#include "Exceptions.h"
#include "Movie.h"
```

```
#include <cstdio>
```

Include dependency graph for CSVWatchlistTests.cpp:



Classes

- class CSVWatchlistTests

Functions

- **TEST_F** (CSVWatchlistTests, test_add)
- **TEST_F** (CSVWatchlistTests, test_update)
- **TEST_F** (CSVWatchlistTests, test_remove)
- **TEST_F** (CSVWatchlistTests, test_find)

6.20.1 Detailed Description

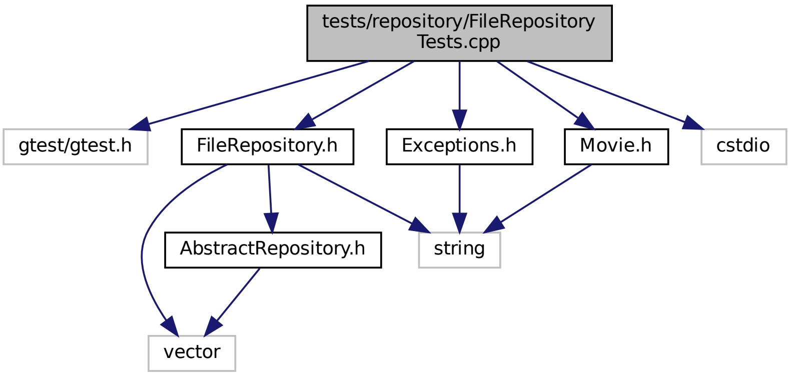
this file contains the tests for the repository

6.21 tests/repository/FileRepositoryTests.cpp File Reference

```
#include "gtest/gtest.h"
#include "FileRepository.h"
#include "Exceptions.h"
#include "Movie.h"
```

```
#include <cstdio>
```

Include dependency graph for FileRepositoryTests.cpp:



Classes

- class FileRepositoryTests

Functions

- **TEST_F** (FileRepositoryTests, test_add)
- **TEST_F** (FileRepositoryTests, test_update)
- **TEST_F** (FileRepositoryTests, test_remove)
- **TEST_F** (FileRepositoryTests, test_find)

6.21.1 Detailed Description

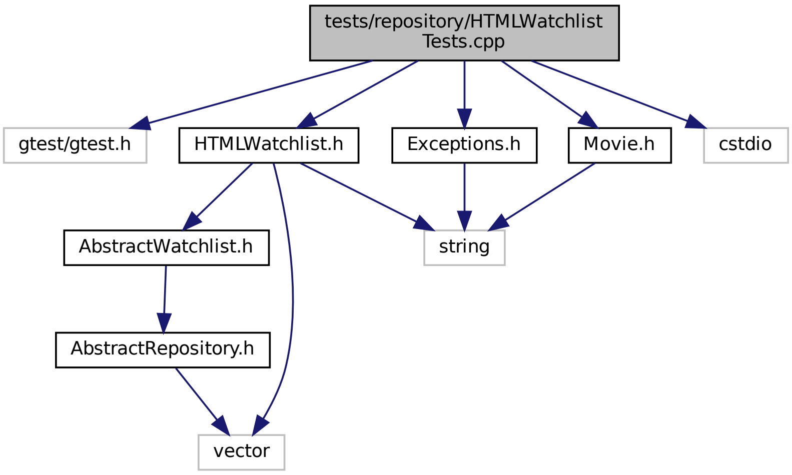
this file contains the tests for the repository

6.22 tests/repository/HTMLWatchlistTests.cpp File Reference

```
#include "gtest/gtest.h"
#include "HTMLWatchlist.h"
#include "Exceptions.h"
#include "Movie.h"
```

```
#include <cstdio>
```

Include dependency graph for HTMLWatchlistTests.cpp:



Classes

- class HTMLWatchlistTests

Functions

- **TEST_F** (HTMLWatchlistTests, test_add)
- **TEST_F** (HTMLWatchlistTests, test_update)
- **TEST_F** (HTMLWatchlistTests, test_remove)
- **TEST_F** (HTMLWatchlistTests, test_find)

6.22.1 Detailed Description

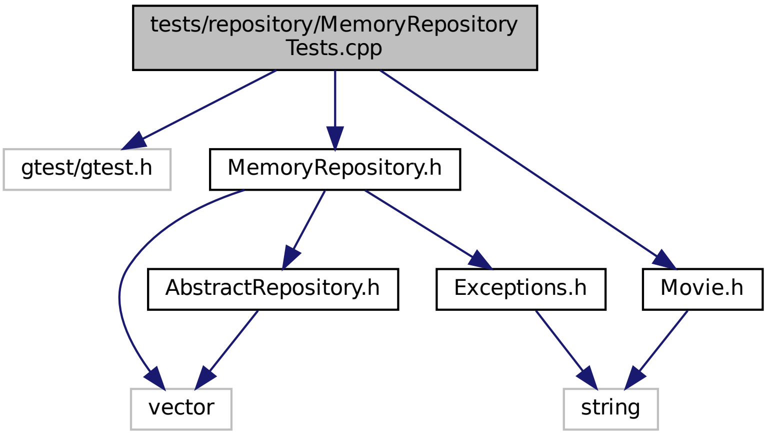
this file contains the tests for the repository

6.23 tests/repository/MemoryRepositoryTests.cpp File Reference

```
#include "gtest/gtest.h"
#include "MemoryRepository.h"
```

```
#include "Movie.h"
```

Include dependency graph for MemoryRepositoryTests.cpp:



Classes

- class MemoryRepositoryTests

Functions

- **TEST_F** (MemoryRepositoryTests, test_add)
- **TEST_F** (MemoryRepositoryTests, test_update)
- **TEST_F** (MemoryRepositoryTests, test_remove)
- **TEST_F** (MemoryRepositoryTests, test_find)

6.23.1 Detailed Description

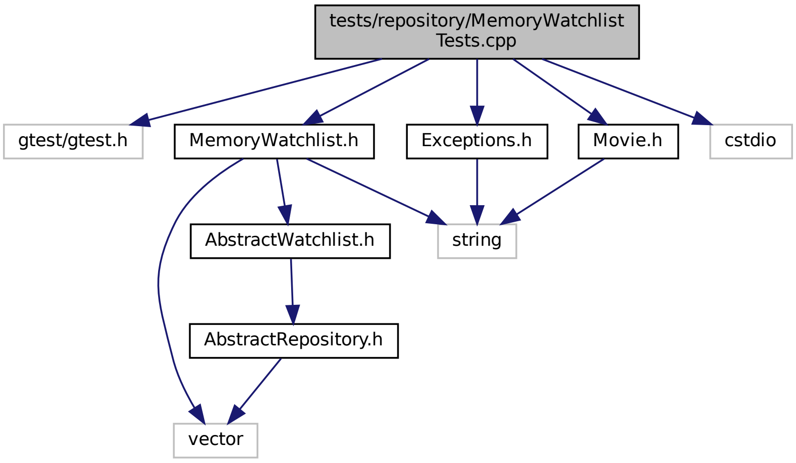
this file contains the tests for the repository

6.24 tests/repository/MemoryWatchlistTests.cpp File Reference

```
#include "gtest/gtest.h"
#include "MemoryWatchlist.h"
#include "Exceptions.h"
#include "Movie.h"
```

```
#include <cstdio>
```

Include dependency graph for MemoryWatchlistTests.cpp:



Classes

- class MemoryWatchlistTests

Functions

- **TEST_F** (MemoryWatchlistTests, test_add)
- **TEST_F** (MemoryWatchlistTests, test_update)
- **TEST_F** (MemoryWatchlistTests, test_remove)
- **TEST_F** (MemoryWatchlistTests, test_find)

6.24.1 Detailed Description

this file contains the tests for the repository

Index