Delft University of Technology

Faculty Electrical Engineering, Mathematics and Computer Science

Managing XML Database with eXist
Web Data Management (IN4331)
2014

Bouke Nederstigt	4008812
Abhishek Sen	4319850

Contents

List of Figures

1 Introduction

For this project we chose to investigate the use of eXist as an XML database. This report gives an overview of the projects and that were conducted and their results. Code excerpts will be displayed throughout the report but anything too big to display here can be found on the following GitHub repository.

https://github.com/bouke-nederstigt/webdatamanagement

2 XPATH and XQUERY Experiment Results

2.1 XPATH Query Output

Title
```
/movies//title
```

All movie titles
```
/movies//title/text()
```

Title of the movies published after 2000
```
/movies/movie[year¿2000]/title/text()
```

Summary of ”Spider-man”
```
/movies/movie[title=”Spider-Man”]/summary/text()
```

Who is the director of Heat?
concat(/movies/movie[title=”Heat”]/director/firstname/text()

Title of the movies featuring Kirsten Dunst
/movies/movie[actor/firstname=”Kirsten” and actor/lastname=”Dunst”]/title/text()

Which movies have a summary?
/movies/movie[summary]/title/text()

Which movies do not have a summary?
/movies/movie[not(summary)]/title/text()

Titles of movies published more than 5 years ago
/movies/movie[year¡2009]/title/text()

What was the role of Clint Eastwoord in Unforgiven?
/movies/movie[title=”Unforgiven”]/actor[firstname=”Clint” and lastname=”Eastwood”]/role/text()

What is the last movie of the document?
/movies/movie[last()]

Title of the film that immediatly precedes Marie Antoinette in the document?

/movies/movie[title=”Marie Antoinette”]/preceding-sibling::*[1]/title/text()

Get the movies whose title contains a ”V”
/movies/movie[contains(title, ”V”)]/title/text()

Get the movies whose cast consist of exactly three actors
/movies/movie[count(actor) = 3]

2.2 XQUERY Output

List the movies published after 2002, including their title and year

```
<results> {
let $ms := doc("movies/movies_alone.xml"), $as := doc("movies/artists_alone.xml")
for $a in $ms//movie where $a/year > 2002
return
<result> {$a/title} {$a/year} </result>
} </results>
```

Create a flat list of all title-role pairs, enclosed in a ”result element

```
<results>{
let $ms := doc("movies/movies_alone.xml"),
$as := doc("movies/artists_alone.xml")
for $t in $ms//movie
return
    for $a in $t//actor
    return
        <result>
            {$t/title}
            <role>{string($a/@role)}</role>
        </result>
}
</results>
```

Give the title of movies where the director is also one of the actors

```
let $ms := doc("movies/movies_alone.xml"),
$as := doc("movies/artists_alone.xml")
for $t in $ms//movie
where $t/director/@id = $t/actor/@id
return
$t/title/text()
```

Show the movies, grouped by genre

```
<results>{
let $ms := doc("movies/movies_alone.xml"),
$as := doc("movies/artists_alone.xml")
for $genre in distinct-values($ms//movie/genre)
let $t := $ms//movie[genre=$genre]
return
    <result> <genre> {$genre} </genre> {$t} </result>
}</results>
```

For each distinct actor's id, show the titles of the movies where this actor plays a role

```
<results>{
let $ms := doc("movies/movies_alone.xml"),
$as := doc("movies/artists_alone.xml")
for $actorId in distinct-values($ms//movie/actor/@id)
let $t := $ms//movie[actor/@id = $actorId]/title
return
    <actor> {$actorId}, {$t} </actor>
}</results>
```

Variant only showing actors playing in at least two movies

```
<results>{
let $ms := doc("movies/movies_alone.xml"),
$as := doc("movies/artists_alone.xml")
for $actorId in distinct-values($ms//movie/actor/@id)
let $t := $ms//movie[actor/@id = $actorId]/title
return
    if(count($t) > 1) then
        <actor> {$actorId}, {$t} </actor>
    else
        ()
}</results>
```

Give the title of each movie, along with the name if its director

```
<results>{
let $ms := doc("movies/movies_alone.xml"),
$as := doc("movies/artists_alone.xml")
for $movie in $ms//movie, $director in $as//artist[@id = $movie/director/@id]
return
    <result>
        {$movie/title}
        <director>
            {$director/first_name}
            {$director/last_name}
        </director>
    </result>
}</results>
```

Give the title of each movie, and a nested element giving the list of actors with their role

```
<results>{
let $ms := doc("movies/movies_alone.xml"),
$as := doc("movies/artists_alone.xml")
for $movie in $ms//movie
return
    <result>
        {$movie/title}
        <actors> {
        for $a in $movie/actor, $actor in $as//artist[@id = $a/@id]
     return
       <actor>
           {$actor/first_name}
           {$actor/last_name}
           <role>{string($a/@role)}</role>
       </actor>
     } </actors>
    </result>
}</results>
```

For each movie that has at least two actors, list the title and firs two actors, and an empty

```
<results>{
let $ms := doc("movies/movies_alone.xml"), $as := doc("movies/artists_alone.xml")
for $movie in $ms//movie
let $count := count($movie/actor)
return
    if($count > 1) then
        <result>
            {$movie/title} {
             for $a in subsequence($movie/actor, 1, 2), $actor in $as//artist[@id =
         return
               <actor>
               {$actor/first_name/text()} {$actor/last_name/text()} as {string($a/
               </actor>
         }
         {if($count > 2) then
             <et-al/>
         else
             ()
         }
```

```
            </result>
        else
            ()
    }</results>
```

List the titles and years of all movies directed by Clint Eastwood after 1990, in alphabetical

```
    <results>{
    let $ms := doc("movies/movies_alone.xml"),
    $as := doc("movies/artists_alone.xml")
    for $actor in $as//artist[first_name="Clint" and last_name="Eastwood"]
        for $movie in $ms//movie[director/@id = $actor//@id]
        where $movie/year > 1990
        order by $movie/title
        return
            <result>
                {$movie/title}
                {$movie/year}
            </result>
    }</results>
```

3 Movie Application

For this part of the project we were asked to implement a simple movie application lookup from an existing movie XML database. Figure 1 below shows a screenshot of the movie application running on the eXist environment. The application was written in the eXist IDE environment and all the code for this application can be found on GitHub link.

3.1 Search Criteria

The search bar for the sample application allows users to search movies with the following criteria:

- All
- Title
- Keywords

- Year
- Director
- Actor
- Genre

3.2 Selecting Movie From Returned Results

If a matching movie(s) is found, a list is returned. More details about the movie can be found if the user clicks on the movie as shown in Figure 2.

4 Shakespeare Application

The purpose of the application is to be able to browse through a Shakespeare play in order to analyze its content, read some specific parts and maybe find related information. To do so the application will consist of the following components. The application was written in the eXist IDE environment and all the code for this application can be found on GitHub link from the introduction. Figure 3 shows a screenshot of the Shakespeare application with the available plays listed in order.

4.1 Architecture

The architecture for the Shakespeare is detailed below:

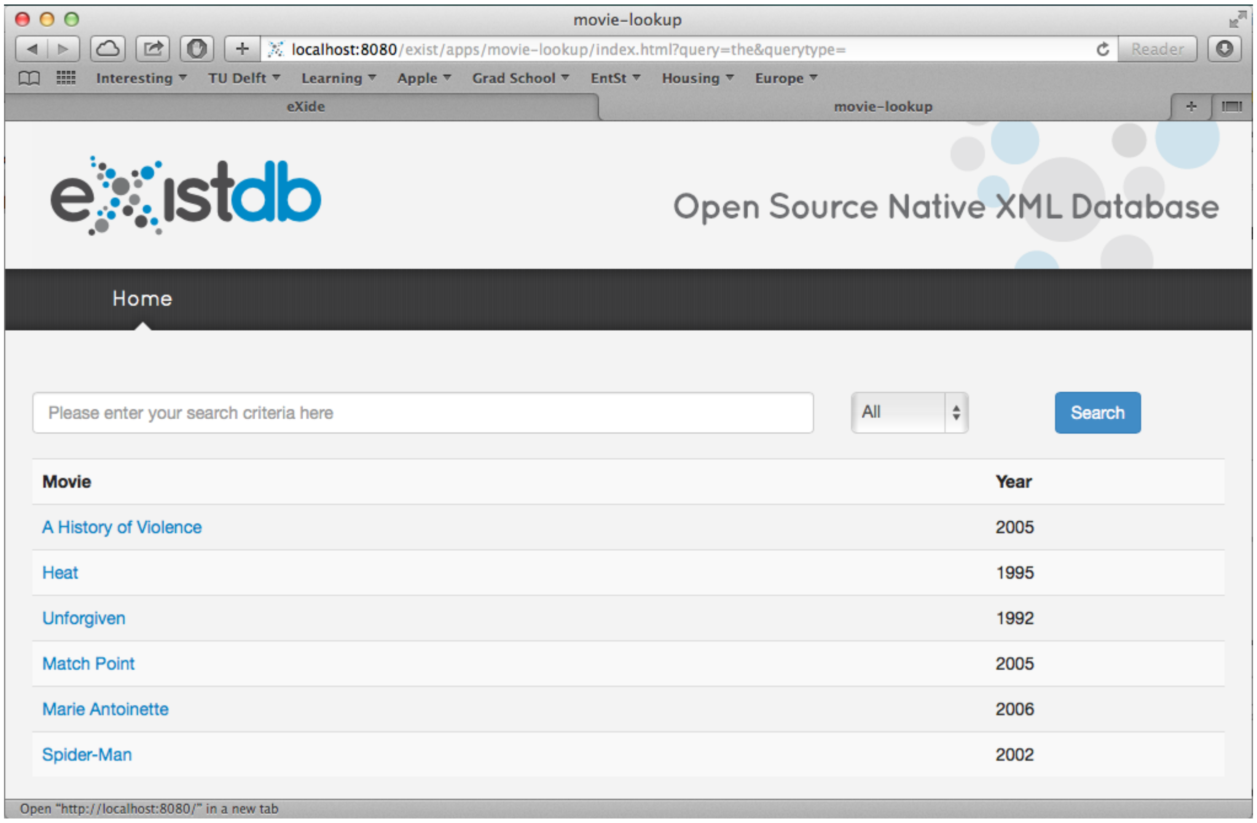
Pages (a) Overview of all plays, linking to contents of play

  i. Contents of play, includes organization in acts and scenes and characters present in scene

  ii. Displaying characters and their parts per act/scene

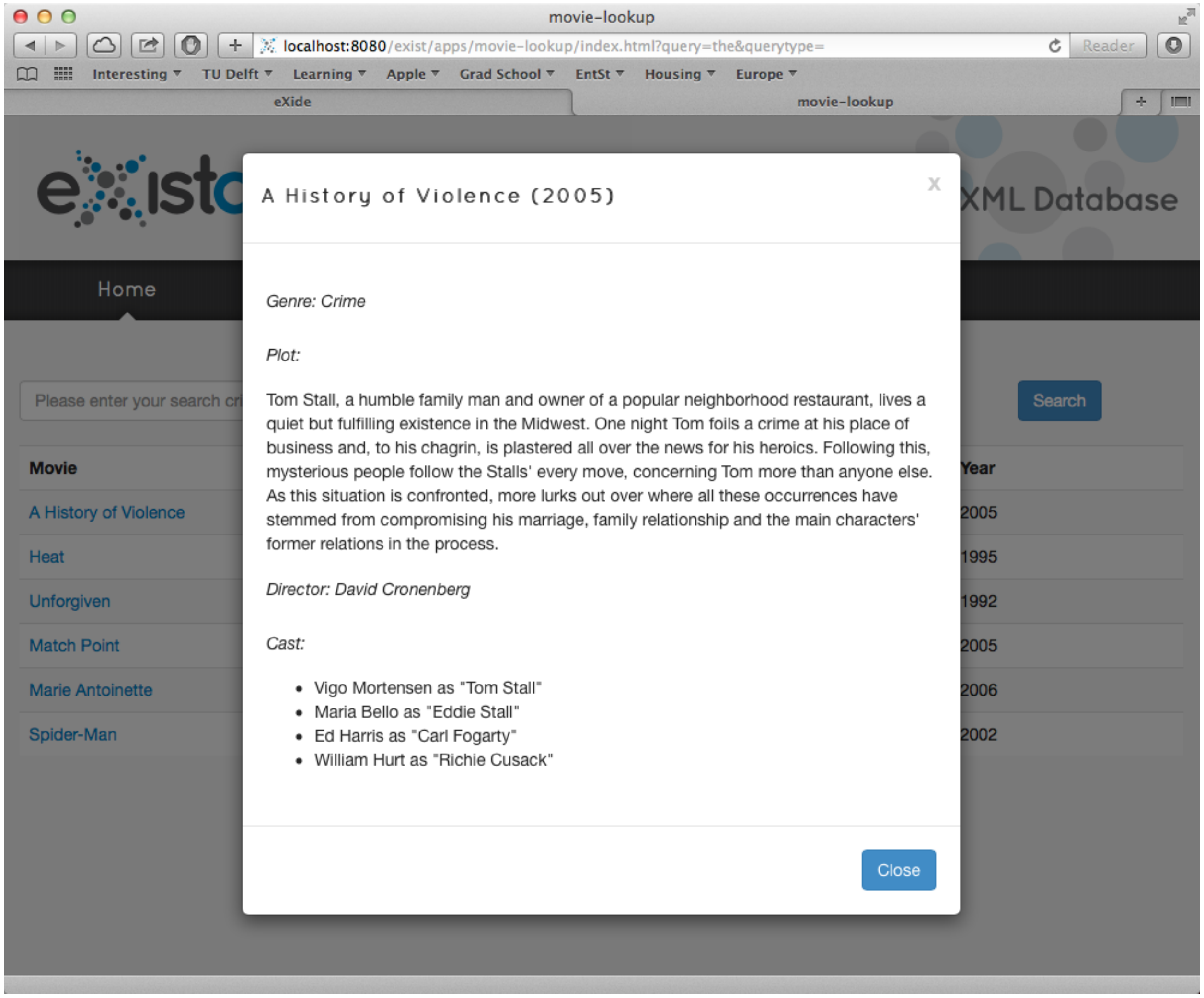
1. Navigation handler - Decides which page to return based on URL
2. All scenes and character references appearing link to their respective pages
3. Navigation Menu
   (a) Should always include link to full summary of current play
   (b) Link to see play overview

Figure 4 below shows the table of contents for Macbeth. The acts, scenes and characters are listed.


movie-lookup
localhost:8080/exist/apps/movie-lookup/index.html?query=the&querytype=
Reader
Interesting
TU Delft
Learning
Apple
Grad School
EntSt
Housing
Europe
eXide
movie-lookup
existdb
Open Source Native XML Database
Home
Please enter your search criteria here
All
Search
Open "http://localhost:8080/" in a new tab

Movie	Year
A History of Violence	2005
Heat	1995
Unforgiven	1992
Match Point	2005
Marie Antoinette	2006
Spider-Man	2002

Figure 1: Screenshot of Movie Application



A History of Violence (2005)

Genre: Crime

Plot:

Tom Stall, a humble family man and owner of a popular neighborhood restaurant, lives a quiet but fulfilling existence in the Midwest. One night Tom foils a crime at his place of business and, to his chagrin, is plastered all over the news for his heroics. Following this, mysterious people follow the Stalls' every move, concerning Tom more than anyone else. As this situation is confronted, more lurks out over where all these occurrences have stemmed from compromising his marriage, family relationship and the main characters' former relations in the process.

Director: David Cronenberg

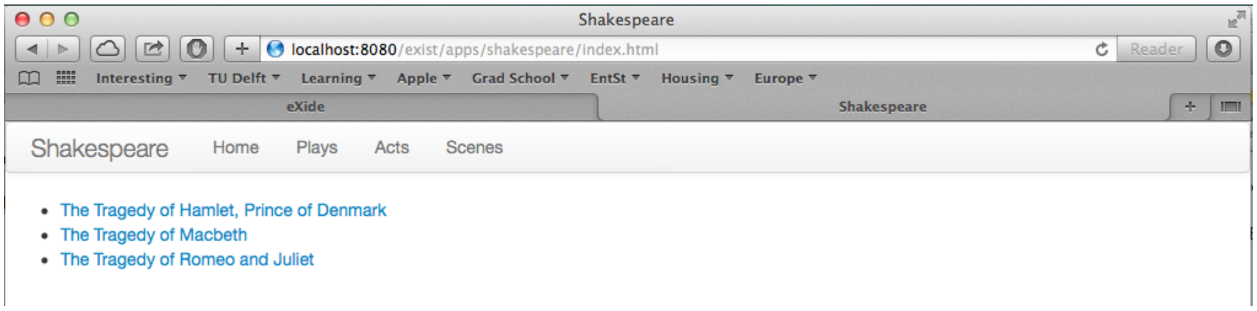
Cast:

- Vigo Mortensen as "Tom Stall"
- Maria Bello as "Eddie Stall"
- Ed Harris as "Carl Fogarty"
- William Hurt as "Richie Cusack"

Close

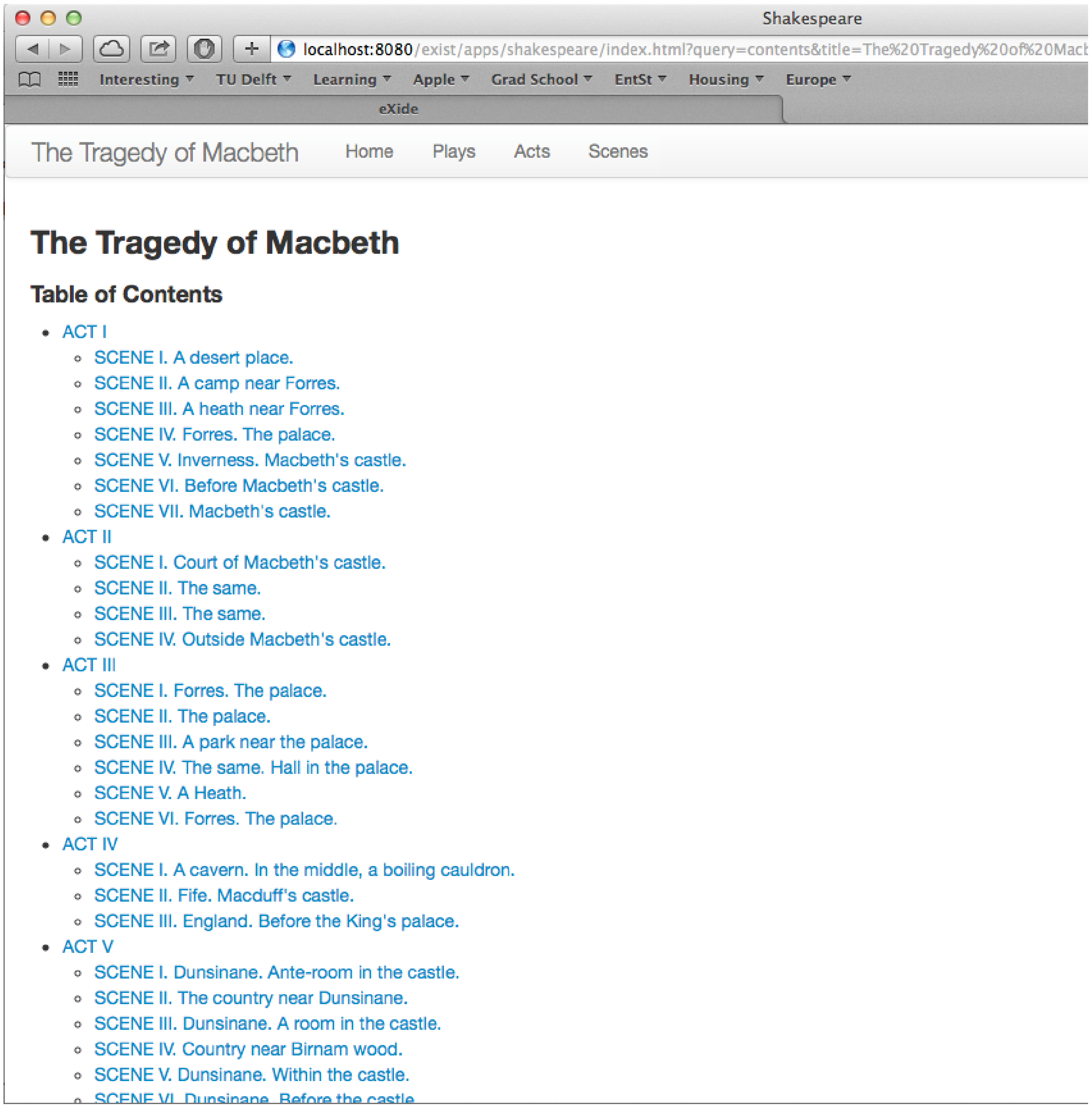
Movie	Year
A History of Violence	2005
Heat	1995
Unforgiven	1992
Match Point	2005
Marie Antoinette	2006
Spider-Man	2002

Figure 2: Additional Movie Information



- The Tragedy of Hamlet, Prince of Denmark
- The Tragedy of Macbeth
- The Tragedy of Romeo and Juliet

Figure 3: Screenshot of Shakespeare Application



The Tragedy of Macbeth

Table of Contents

- ACT I
  - SCENE I. A desert place.
  - SCENE II. A camp near Forres.
  - SCENE III. A heath near Forres.
  - SCENE IV. Forres. The palace.
  - SCENE V. Inverness. Macbeth's castle.
  - SCENE VI. Before Macbeth's castle.
  - SCENE VII. Macbeth's castle.
- ACT II
  - SCENE I. Court of Macbeth's castle.
  - SCENE II. The same.
  - SCENE III. The same.
  - SCENE IV. Outside Macbeth's castle.
- ACT III
  - SCENE I. Forres. The palace.
  - SCENE II. The palace.
  - SCENE III. A park near the palace.
  - SCENE IV. The same. Hall in the palace.
  - SCENE V. A Heath.
  - SCENE VI. Forres. The palace.
- ACT IV
  - SCENE I. A cavern. In the middle, a boiling cauldron.
  - SCENE II. Fife. Macduff's castle.
  - SCENE III. England. Before the King's palace.
- ACT V
  - SCENE I. Dunsinane. Ante-room in the castle.
  - SCENE II. The country near Dunsinane.
  - SCENE III. Dunsinane. A room in the castle.
  - SCENE IV. Country near Birnam wood.
  - SCENE V. Dunsinane. Within the castle.
  - SCENE VI. Dunsinane. Before the castle.

Figure 4: Table of Contents for Macbeth