



universidade
de aveiro

University of Aveiro
Computer and Telematics Engineering
Data and Knowledge Engineering



UA Aggregator

André Pinho
Nº mec: 80313

Ana Margarida Silva
Nº mec: 77752

Diego Hernandez
Nº mec: 77013

10th of November of 2018
Aveiro



Index

1.0 Introduction	3
2.0 Schedules	4
2.1 The Datasource	4
2.2 Datasource Manipulation and XSLT	5
2.3 Database Queries	8
2.4 Challenges	9
3.0 Reserve Classroom	11
3.1 The Application	11
3.2 Technology Used	11
3.3 XQueries	13
3.4 Challenges	15
4.0 SAS	16
4.1 The Application	16
4.2 Technology Used	16
4.2 XML Schema	17
5.0 Weather Prediction	18
5.1 The Application	18
5.2 Technology Used	18
6.0 UA Newsletter	19
6.1 The Application	19
6.2 Technology Used	20
6.3 XML Schema	21
7.0 SAC (Serviços Academicos)	22
7.1 The Application	22
7.2 Technology Used	22
7.3 XML Schema	23
7.4 Challenges	23
8.0 Parking Lot	24
8.1 The Application	24
8.2 Technology Used	25
8.3 XMLSchema	26
8.4 Challenges	26
9.0 Conclusion	27
10.0 Installation and Execution	27
10.1 Python dependencies	27
10.2 Execution	27



1.0 Introduction

UA Aggregator is a web based platform created with the first intuition of giving students of University of Aveiro the possibility to create and organize their schedules of the current semester by giving them all the possible schedules, given the year and course that the client has selected.

For the construction of schedules it was given an excel file with information that we could work out for the functionality. The processing of the data contained on the excel will be explained later on. Knowing the diverse open APIs that University of Aveiro has provided by Academic Playground & Innovation, the opportunity was took and it was decided that UA Aggregator will become more than a Schedule Maker. UA Aggregator is now a web platform where students can reserve classrooms for their studies given the selected hours. UA Aggregator is also an application where students can see the weather prediction, the available parking lots at the University, see what on the todays menu of the University's Canteens, know the status of Academic Services, and finally a Newsletter of the University of Aveiro.

All of this was done with Django's framework, BaseX, XML, XML Schema, XQuery and XSLT technology.

Our mindset for this project was to make of use all the advantages of the technology mentioned above to create an application that could be of some use for the University Community.



2.0 Schedules

Gerar horarios

Curso: MIECT ▼ Ano: 4º ▼ Gerar

Dias da Semana	9h	10h	11h	12h	13h	14h	15h	16h	17h	18h	19h	20h
segunda-feira	ARQUITECTURA DE REDES AVANÇADAS [T1] ANF. V		COMPUTAÇÃO VISUAL [T1] ANF. V			ARQUITECTURA DE COMPUTADORES AVANÇADA [P1] 04.1.01		COMPUTAÇÃO VISUAL [P1] 04.1.04				
terça-feira	ARQUITECTURA DE COMPUTADORES AVANÇADA [T1] ANF. V		ENGENHARIA DE DADOS E CONHECIMENTO [T1] ANF. V									
quarta-feira	SEGURANÇA [T1] ANF. V											
quinta-feira	ARQUITECTURA DE REDES AVANÇADAS [P3] 04.2.20		ENGENHARIA DE DADOS E CONHECIMENTO [P2] 04.2.08			SEGURANÇA [P1] 04.2.03						
sexta-feira												

The picture above shows one of the generated schedules for MIECT's 4th year. By scrolling the web page, it's possible to see more.

2.1 The Datasource

Since the university doesn't offer a public API to retrieve the schedules of all courses, we asked Prof. José Vieira for help. Prof. Vieira contacted Cláudio Teixeira from sTIC, who got us an Excel document (stubs/horarios/Horarios_Curso_DET1.xlsx) with all the classes from all weeks of DET1 for the first semester of this current year:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Eventoid	horainicio	horafim	Sala	CodigoUC	NomeUC	NomeTurno	Di	Semana	Tip	Nº	CodigoCu	NomeCurso	An
2	110531	11:00:00	13:00:00	23.3.22 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLAB1	2	2018-10-08 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
3	110531	11:00:00	13:00:00	23.3.22 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLAB1	2	2018-10-22 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
4	110531	11:00:00	13:00:00	23.3.22 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLAB1	2	2018-11-05 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
5	110531	11:00:00	13:00:00	23.3.22 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLAB1	2	2018-11-19 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
6	110531	11:00:00	13:00:00	23.3.22 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLAB1	2	2018-12-03 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
7	110531	11:00:00	13:00:00	23.3.22 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLAB1	2	2018-12-10 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
8	110532	11:00:00	13:00:00	23.3.23 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB1	4	2018-10-08 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
9	110532	11:00:00	13:00:00	23.3.23 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB1	4	2018-10-22 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
10	110532	11:00:00	13:00:00	23.3.23 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB1	4	2018-11-05 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
11	110532	11:00:00	13:00:00	23.3.23 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB1	4	2018-11-19 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
12	110532	11:00:00	13:00:00	23.3.23 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB1	4	2018-12-03 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
13	110532	11:00:00	13:00:00	23.3.23 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB1	4	2018-12-10 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
14	110533	09:00:00	11:00:00	23.3.21 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB2	4	2018-10-08 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
15	110533	09:00:00	11:00:00	23.3.21 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB2	4	2018-10-22 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
16	110533	09:00:00	11:00:00	23.3.21 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB2	4	2018-11-05 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
17	110533	09:00:00	11:00:00	23.3.21 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB2	4	2018-11-19 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
18	110533	09:00:00	11:00:00	23.3.21 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB2	4	2018-12-03 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
19	110533	09:00:00	11:00:00	23.3.21 - Lab. Fis	41791	ELEMENTOS DE FISICA	Grupo III - PLB2	4	2018-12-10 00:00:00.000	P	6	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1
20	110534	11:00:00	13:00:00	ANF, 12.2.1	41791	ELEMENTOS DE FISICA	Grupo III - TPA	3	2018-09-17 00:00:00.000	TP	14	8295	LICENCIATURA EM ENGENHARIA INFORMATICA	1

In the image above there is a very small subset of that Excel.

2.2 Datasource Manipulation and XSLT

This document is not in the format we can work with, so we needed to get a way to extract data. Since the format given is not text, we tried to export to different formats. Firstly we tried “Microsoft Excel 2003 XML”. Half through the process of converting that format to our own database format, we noticed that that format was not as regular as we thought: there were places where not every column was represented (when the column was empty, there was a <Cell> missing, and in the next one, the next cell would be <Cell ss:Index=“next_index”>), making it’s parsing significantly more difficult. After that, we tried “Flat XML ODF Spreadsheet (.fods)”. This format was more regular than the last one, but had unnecessary complexity and since it had namespaces, the xpath would be a lot more verbose and complex. For example a simple xpath (ex: /A/B/C), all elements needed to be converted into name()='atributename' (ex: /*[name()='A']/*[name()='B']/*[name()='C']).

Finally, we tried to export the Excel to XHTML (stubs/horarios/Horarios_Cursos_DET1.html). This filetype worked well, since there were no trouble with namespaces and with non fixed form formats. To transform this file to a usable format (swap xhtml tags, remove duplicate data and rearrange data, etc...), we used XSLT (stubs/horarios/convert.xsd).



XSLT's Input:

```
<td align="left" valign="bottom"><font color="#000000">LICENCIATURA EM ENGENHARIA INFORMÁTICA (1º CICLO)</font></td>
<td align="right" valign="bottom" sdval="1" sdnum="1033;"><font color="#000000">1</font></td>
</tr>
<tr>
<td height="20" align="right" valign="bottom" sdval="110531" sdnum="1033;"><font color="#000000">110531</font></td>
<td align="left" valign="bottom"><font color="#000000">11:00:00.0000000</font></td>
<td align="left" valign="bottom"><font color="#000000">13:00:00.0000000</font></td>
<td align="left" valign="bottom"><font color="#000000">23.3.22 - Lab. Fis</font></td>
<td align="right" valign="bottom" sdval="41791" sdnum="1033;"><font color="#000000">41791</font></td>
<td align="left" valign="bottom"><font color="#000000">ELEMENTOS DE FÍSICA</font></td>
<td align="left" valign="bottom"><font color="#000000">Grupo III - PLAB1</font></td>
<td align="right" valign="bottom" sdval="2" sdnum="1033;"><font color="#000000">2</font></td>
<td align="left" valign="bottom"><font color="#000000">2018-12-03 00:00:00.000</font></td>
<td align="left" valign="bottom"><font color="#000000">P</font></td>
<td align="right" valign="bottom" sdval="6" sdnum="1033;"><font color="#000000">6</font></td>
<td align="right" valign="bottom" sdval="8295" sdnum="1033;"><font color="#000000">8295</font></td>
<td align="left" valign="bottom"><font color="#000000">LICENCIATURA EM ENGENHARIA INFORMÁTICA (1º CICLO)</font></td>
<td align="right" valign="bottom" sdval="1" sdnum="1033;"><font color="#000000">1</font></td>
</tr>
<tr>
<td height="20" align="right" valign="bottom" sdval="110531" sdnum="1033;"><font color="#000000">110531</font></td>
<td align="left" valign="bottom"><font color="#000000">11:00:00.0000000</font></td>
<td align="left" valign="bottom"><font color="#000000">13:00:00.0000000</font></td>
<td align="left" valign="bottom"><font color="#000000">23.3.22 - Lab. Fis</font></td>
<td align="right" valign="bottom" sdval="41791" sdnum="1033;"><font color="#000000">41791</font></td>
<td align="left" valign="bottom"><font color="#000000">ELEMENTOS DE FÍSICA</font></td>
<td align="left" valign="bottom"><font color="#000000">Grupo III - PLAB1</font></td>
<td align="right" valign="bottom" sdval="2" sdnum="1033;"><font color="#000000">2</font></td>
<td align="left" valign="bottom"><font color="#000000">2018-12-10 00:00:00.000</font></td>
<td align="left" valign="bottom"><font color="#000000">P</font></td>
<td align="right" valign="bottom" sdval="6" sdnum="1033;"><font color="#000000">6</font></td>
<td align="right" valign="bottom" sdval="8295" sdnum="1033;"><font color="#000000">8295</font></td>
<td align="left" valign="bottom"><font color="#000000">LICENCIATURA EM ENGENHARIA INFORMÁTICA (1º CICLO)</font></td>
<td align="right" valign="bottom" sdval="1" sdnum="1033;"><font color="#000000">1</font></td>
</tr>
<tr>
<td height="20" align="right" valign="bottom" sdval="110532" sdnum="1033;"><font color="#000000">110532</font></td>
<td align="left" valign="bottom"><font color="#000000">11:00:00.0000000</font></td>
```



XSLT's Output:

```
pinho@ce:~$ cat /dev/null
</horarios>
</turma>
<turma turno="OT" tipo="OT">
  <horarios>
    <aula dia_da_semana="segunda-feira">
      <sala>10.1.11</sala>
      <inicio>18:00:00.0000000</inicio>
      <fim>19:00:00.0000000</fim>
    </aula>
  </horarios>
</turma>
</turmas>
</cadeira>
</cadeiras>
</curso>
<curso codigo="8240">
  <nome>MESTRADO INTEGRADO EM ENGENHARIA DE COMPUTADORES E TELEMÁTICA</nome>
  <cadeiras>
    <cadeira codigo="47121">
      <nome>GESTÃO DE PROJECTOS E EMPREENDEDORISMO</nome>
      <ano>5</ano>
      <turmas>
        <turma turno="P 1 - MIECT" tipo="P">
          <horarios>
            <aula dia_da_semana="segunda-feira">
              <sala>10.1.8</sala>
              <inicio>09:00:00.0000000</inicio>
              <fim>11:00:00.0000000</fim>
            </aula>
          </horarios>
        </turma>
        <turma turno="P 2 - MIECT" tipo="P">
          <horarios>
            <aula dia_da_semana="segunda-feira">
              <sala>10.1.8</sala>
              <inicio>09:00:00.0000000</inicio>
              <fim>11:00:00.0000000</fim>
            </aula>
          </horarios>
        </turma>
        <turma turno="TP 1 - MIECT" tipo="TP">
          <horarios>
            <aula dia_da_semana="segunda-feira">
              <sala>10.1.8</sala>
```

Since we needed 'distinct-values' function on XPath, and since the lxml library does not support XPath 3.0 and XSLT 3.0, we used BaseX (baseX/) with custom libraries (SaxonHE and Xerces2) to support these features.

Finally, we made a script in python (stubs/horarios/converter.py) that calls BaseX's HTMLParser and XSLT Transform and prints to the terminal the transformed XML. The final XML is used as a database (horarios_database) and it's read-only.

2.3 Database Queries

A temporary database is created when generating schedules for storing all possible combinations. On the file "stubs/horarios/generate_horarios.xq", several UDFs were defined to help us construct the schedules, serving the purposes of:

- Creating/deleting schedule database
- Creating/deleting schedule combination
- Appending subject and classes to schedule combination
- Counting schedule combinations
- Getting current status of schedule combinations
- Verifying if a schedule is valid without class overlap (depicted below)

```
declare function local:fits_horario($id as xs:integer, $suc as xs:integer, $turma as xs:string) as item() {
  let $new_hours := local:get_turma_hours($suc, $turma)
  let $cnt:=count(
    for $new in $new_hours//aula
    return(
      for $aula in doc('self_horarios')//aula[../../../../../../../../@id=$id]
      where $aula/@dia da semana=$new/@dia da semana and
      ($new/inicio<=$aula/inicio and $new/fim>$aula/inicio or
       $new/inicio>$aula/inicio and $new/inicio<$aula/fim)
      return <unmatch/>))
  return $cnt=0
};
```

Besides those functions, a few more were created to filter data or simplify the previous ones. Those are used to:

- Get a class
- Get a subject
- Get a given class' hours
- Get all subjects from a given course and year
- Get classes from a given subject that belongs to a specific type (e.g: practical class)

2.4 Challenges

Due to the algorithmic complexity of schedule generation (it's a NP-Hard problem), we simplified it by doing the combinatorial process on Python. To do so, on stubs/horarios/horario.py there are several functions to handle this issue. Besides combine_courses(), which takes care of the combinatorial process, two sets of defs (python functions) were created to translate XML to Python structures as can be seen below. On course_to_xml(), the UDFs concerning the schedule generation are called. The assertion that a certain combination of classes is valid is also made there by calling the fits_horario() UDF.

```
def xml_to_courses(ucs):
    courses = ucs.findall("./cadeira")
    entries = dict()
    for c in courses:
        code = c.attrib['codigo']
        entries[code] = xml_to_course(c)

    return entries

def xml_to_course(course):
    tipos = tipo_aulas_uc(course)
    turmas = course.findall("./turma")
    struct = dict()

    for t in tipos:
        if t != "OT":
            struct[t] = []

    for t in turmas:
        if t.attrib["tipo"] != "OT":
            if len(struct[t.attrib["tipo"]]) < 4:
                struct[t.attrib["tipo"]].append(t.attrib["turno"])

def courses_to_xml(courses, curso_id):
    i = 0
    count = 0
    for c in courses:
        call_gerar_horarios('local:create_option(' + str(i) + ')')

        possible = course_to_xml(c, i, curso_id)
        if possible:
            count+=1
            if count > 7:
                break;
        i+=1

def course_to_xml(course, i, curso_id):
    del_option = False
    for c in course:
        uc = c[0]

        call_gerar_horarios('local:append_cadeira_step1(' + str(i) + ', ' + str(curso_id) + ', ' + str(uc) + ')')
        call_gerar_horarios('local:append_cadeira_step2(' + str(i) + ', ' + str(uc) + ')')

        for e in c[1]:
            turma_val = e
            turma_xml = call_gerar_horarios('local:get_turma(' + str(uc) + ', ' + str(curso_id) + ', \'\' + \
                str(turma_val) + '\')')
            flag = call_gerar_horarios('local:fits_horario(' + str(i) + ', ' + str(uc) + ', \'\' + \
                str(etree.fromstring(turma_xml).attrib['turno']) + '\')')
            print(flag)
            if flag == "true":
                call_gerar_horarios('local:append_turma(' + str(i) + ', ' + str(curso_id) + ', ' + \
                    str(uc) + ', \'\' + str(etree.fromstring(turma_xml).attrib['turno']) + '\')')
            else:
                del_option = True
                break;
        if del_option:
            call_gerar_horarios('local:del_option(' + str(i) + ')')
            break
    return not del_option
```

Since for each combination of classes it's necessary to verify its validity and to do so the UDF is called, the generation process is not very optimized and may take a very long time to complete (hundreds of valid schedules in a space of tens of thousands). In order to make the application usable, some aspects had to be simplified.



- Instead of generating all possible schedule outcomes, the generation is considered complete after 8 valid schedules are found.
- Some subjects have over 6 different classes, which makes generation more time consuming, so there's a maximum of 4 classes that will be taken into account.

Some problems arise with our choices:

- Since we're generating schedules based on course/year instead of processing them individually, this restriction can make it impossible to generate schedules in years where there are optional subjects as they can overlap.
- By reducing the number of allowed classes per subject type to 4 and given the fact that the chosen set of classes is random, it may be impossible to generate schedules for 1st year subjects as the pool of classes may overlap nevertheless.

Regarding the XSLT, even after finding a good source of data (XHTML) there was still a lot of setbacks. Since the data was duplicated (for each class, the information was duplicated for each week), there was needed some research on how to iterate on the unique entries. Secondly, since the data was in list form, the XSLT was not linear. In other words, inner loops of the XSLT sometimes needed to get information from other entries from the same level, or above.



3.0 Reserve Classroom

3.1 The Application

Procurar Salas Disponíveis

Data :

Hora Inicial :

Hora Final :

Salas livres para 2018-11-20, das 10:00 até as 11:00

- ☐ 23.3.22 - Lab. Fis
- ☐ 23.3.23 - Lab. Fis
- ☐ 23.3.21 - Lab. Fis
- ☐ ANF. 12.2.1
- ☐ 28.02.23
- ☒ ANF. 23.1.6
- ☐ 23.3.4
- ☐ 23.3.15
- ☐ 23.3.10
- ☐ 04.2.07
- ☐ 04.1.19
- ☐ 04.2.11
- ☐ 04.1.01
- ☐ ANF. 11.1.3
- ☐ 04.1.03
- ☐ 23.2.14
- ☐ ANF. 11.1.10
- ☐ 10.3.14
- ☐ 10.3.6

On this section the user can reserve an available classroom by choosing the date and the between times of the usage of an available classroom. Upon having this information filled, the user can search the available classrooms and select the desired one. The user must fill his mecanograph number so the reservation is on his behalf. After submitting the requesting information, the classroom will be reserved for the user and won't be available to reserve on that day on top, or inside of this time interval. All the data submitted will be stored on a Basex Database (reservas_database, described below).

3.2 Technology Used

Since we have the information of all courses of DETI, one can deduce from it all the classrooms that exist, and which ones are available at a certain time. We used that particularity to our



advantage. So, using the same database that we use to generate schedules, we query (on demand) for empty rooms. At the same time, we query a second database (reservas_database) where we store the rooms that already were reserved:

```
<reserva>
  <entrada nmec="80313" sala="04.1.02" dia="2018-11-22" inicio="14:00:00" fim="15:15:00"/>
  <entrada nmec="77752" sala="04.1.03" dia="2018-11-30" inicio="20:00:00" fim="21:30:00"/>
  <entrada nmec="77013" sala="04.1.01" dia="2018-12-10" inicio="10:00:00" fim="11:00:00"/>
</reserva>
```

Salas livres para 2018-12-10, das 10:00 até as 11:00

- ☐ 23.3.22 - Lab. Fis
- ☐ 23.3.21 - Lab. Fis
- ☐ ANF. 12.2.1
- ☐ 28.02.23
- ☐ ANF. 23.1.6
- ☐ 23.3.4
- ☐ 23.3.10
- ☐ 04.2.16
- ☒ 04.1.01
- ☐ ANF. 11.1.3
- ☐ 04.1.03
- ☐ 23.2.13
- ☐ 23.2.7

Salas livres para 2018-12-01, das 10:30 até as 12:00

- ☐ 23.3.22 - Lab. Fis
- ☐ 23.3.23 - Lab. Fis
- ☐ 23.3.21 - Lab. Fis
- ☐ ANF. 12.2.1
- ☐ 28.02.23
- ☐ ANF. 23.1.6
- ☐ 23.3.4
- ☐ 23.3.15
- ☐ 23.3.10
- ☐ 04.2.16
- ☐ 04.2.07
- ☐ 04.1.02
- ☐ 04.1.19
- ☐ 04.2.25
- ☐ 04.1.04

The image on the right and on top was taken after the operation of the image on the left was made. On the top image are all the database entries. After reserving a classroom, that same room is not available when there are collisions with other reservations or normal classes.

When the user asks for available classrooms a POST request is made, and with the input given (date and hours) the view will make of use of the module "listar_salas_livre" from the python file horarios.py where xqueries are called, letting the display of the available rooms happen.

After the selection of an available classroom and the submission of the user's mecanograph number, the room view will also make of use another module from the horarior.py, "reservar_sala", creating a XMLQuery to store information about the reservation on the database.



We used Django Framework to display the data from the file views.py. Technologies and languages were used, such as HTML, CSS, Javascript and Bootstrap.
The correspondent template file for this application has the name of room.html.

3.3 XQueries

All XQueries were implemented using User Defined Functions (stubs/horarios/horariofunc.xq).
The most important queries are:

- **get_salas:** Xquery returns all classrooms from database “horarios_database”

```
declare function local:get_salas() as item(){
  <salas>{
    for $sala in distinct-values(doc('horarios_database')//sala)
    return <sala>{$sala}</sala>
  }</salas>
};
```

- **is_sala_sem_aula:** Xquery returns boolean True if a given classroom is not class on the given time interval: the initial and final time and date.

```
declare function local:is_sala_sem_aula($sala as xs:string, $inicio as xs:time, $fim as xs:time, $diadasemana as xs:string) as xs:boolean{
  let $cnt:=count(
    for $aula in doc('horarios_database')//aula
    where $aula/@dia_da_semana=$diadasemana and
          $aula/sala=$sala and
          ($inicio<=$aula/inicio and $fim>$aula/inicio or
           $inicio>$aula/inicio and $inicio<$aula/fim)
    return <unmatch/>)
  return $cnt=0
};
```

- **sala_reservada:** Xquery returns boolean True if the classroom is occupied by a registration on the given day and hours, on the “reservas_database” database.


```
declare function local:sala_reservada($sala as xs:string,$inicio as xs:time,$fim as xs:time,$dia as xs:date) as item(){
  let $cnt:=count(
    for $sent in doc('reservas_database')//entrada[@dia=xs:string($dia) and @sala=$sala]
  )
  where $inicio<=$sent/@inicio and $fim>$sent/@inicio or
        $inicio>$sent/@inicio and $inicio<$sent/@fim
  return <unmatch/>
return $cnt!=0
};
```

- **reservar_sala:** given the mecanograph number, the initial and final hour, the date and the classroom, it will be inserted on the database “reservas_database” the following information, with the purpose of reserving the classroom for the user with the given mecanograph number.

```
declare updating function local:reservar_sala($nmec as xs:integer,$sala as xs:string,$inicio as xs:time,$fim as xs:time,$dia as xs:date){
  if(local:is_sala_sem_aula($sala,$inicio,$fim,functx:day-of-week-pt($dia)) and local:sala_reservada($sala,$inicio,$fim,$dia)=(0=1))
  then(
    (:free spot:)
    insert node <entrada nmec="{ $nmec}" sala="{ $sala}" dia="{ $dia}" inicio="{ $inicio}" fim="{ $fim}" /> into doc('reservas_database')/reserva
  )
};
```

- **get_salas_livres:** given the initial hour, the last hour and the date, this XQuery will return the available classrooms, with the XML format <sala>sala</sala>

```
declare function local:get_salas_livres($inicio as xs:time,$fim as xs:time,$dia as xs:date) as item(){
  <salas>{
    for $sala in local:get_salas()/sala/text()
    where local:is_sala_sem_aula($sala,$inicio,$fim,functx:day-of-week-pt($dia)) and local:sala_reservada($sala,$inicio,$fim,$dia)=(0=1)
    return <sala>{$sala}</sala>
  }</salas>
};
```

- **get_reservas:** xquery returns all the reservations fetched on “reservas_database” database for a student on a certain day;

```
declare function local:get_reservas($nmec as xs:integer,$dia as xs:date) as item(){
  <reservas>{
    for $reserva in doc('reservas_database')//entrada[@nmec=$nmec and @dia=$dia]
    return $reserva
  }</reservas>
};
```



3.4 Challenges

The main difficulty in this section was implementing a restriction based search and related logic in user defined functions of xquery.



4.0 SAS

4.1 The Application

Jantar	Almoço
Almoço	
Refeitório de Santiago	Refeitório do Crasto
Sopa : Sopa de lentilhas	Sopa : Sopa de lentilhas
Prato normal carne : Perna de peru estufada com cenoura e massa esparguete	Prato normal peixe : Meia desfeita de bacalhau (batata cozida e grão de bico)
Prato dieta : Bife de peru grelhado com batata cozida e cenoura	Prato dieta : Tofu à moda de braga
Prato vegetariano : Tofu à moda de braga	Prato opção : Fruta da época ou doce
Prato opção : Bolos de bacalhau com arroz de tomate	Salada : Buffet de saladas
Salada : Buffet de saladas	Diversos : Vitela à lãfões e batata cozida
Diversos : Pão de mistura	Sobremesa : Pão de mistura
Sobremesa : Fruta da época ou doce	
Snack-Bar/Self	
Sopa : Sopa de lentilhas	
Diversos : Tofu à moda de braga	
Sobremesa : Fruta da época ou doce	

On this section the user is able to see the today's lunch and meal at the University of Aveiro Canteens. They are able to see all the available dishes as well.

On the Web application the user is capable to see specifically what it wants to see, the food and canteens served on lunch time and/or meal time, by clicking on the buttons illustrated above on Figure.

4.2 Technology Used

The data is gathered from an API of Academic Playground & Innovation, from the url: <http://services.web.ua.pt/sas/ementas>. The data provided is on XML Format.

The following give us data about the available canteens at University of Aveiro that serves food on meal and/or lunch time, following their corresponding dishes.

The python class responsible to fetch the data about the the SAS service is the SASService from the file uaservices.py. To notice that this class makes of use the abstract class XMLService. This data is fetch by the use of the library lxml. All the xml data fetch is validated by the use of the XMLSchema SASSchema.xsd. The data, if validated, will be structured in a python dictionary format with the purpose of easy manipulation of data to process and display it with Django Framework.

We used Django Framework to display the data from the file views.py. Technologies and languages were used, such as HTML, CSS, Javascript and Bootstrap.

The correspondent template file for this application has the name of canteen.html.




4.2 XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="result">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="menus">
          <xs:complexType>
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element name="menu">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="items">
                      <xs:complexType>
                        <xs:sequence minOccurs="0" maxOccurs="unbounded">
                          <xs:element name="item">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:string">
                                  <xs:attribute name="name" type="xs:string" use="required"/>
                                  <xs:anyAttribute processContents="skip"/>
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                        </xs:sequence>
                      <xs:anyAttribute processContents="skip"/>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              <xs:attribute name="canteen" type="xs:string" use="required"/>
              <xs:attribute name="meal" type="xs:string" use="required"/>
              <xs:attribute name="weekday" type="xs:string" use="required"/>
              <xs:attribute name="disabled" type="xs:string" use="required"/>
              <xs:anyAttribute processContents="skip"/>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      <xs:anyAttribute processContents="skip"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

5.0 Weather Prediction

5.1 The Application

Weather

	<i>Monday, November 12</i> Clear Sky Min: 6°C Max: -- Humidity: 89%
	<i>Tuesday, November 13</i> Sunny Min: 4°C Max: 17°C Humidity: 86%
	<i>Wednesday, November 14</i> Sunny Min: 5°C Max: 20°C Humidity: 87%

On this section the user is able to see the prediction for the upcoming 3 days in Aveiro. The data that is displayed is the date, status of the sky, minimum and maximum temperature on Celsius, and humidity percentage.

Corresponding to the overall status of the weather for each day it will be displayed a self illustrative image of the status of the corresponding day weather. For example, if the weather is sunny, the web application will display an image of a sun on the corresponding day section.

The web application will only gather information about the weather for the next 3 upcoming days.

5.2 Technology Used

The information about the weather is provided by a RSS feed accessible in: <https://weather-broker-cdn.api.bbci.co.uk/en/forecast/rss/3day/2742611>.

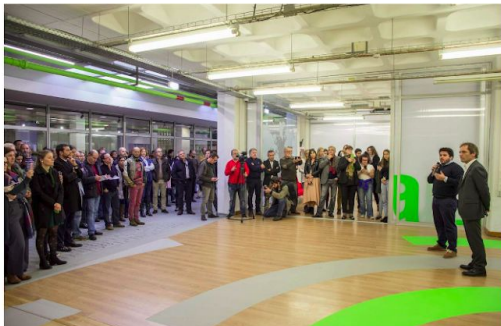
The following give us data about the Weather of Aveiro on the upcoming 3 days.

The class responsible to fetch the data about the Weather in Aveiro is the WeatherService from the file uaservices.py. To notice that this class makes use of the abstract class XMLService. This data is fetched by the library lxml. The data can be structured in a python dictionary format with the purpose of easy manipulation of data to process and display it with Django Framework. We used Django Framework to display the data from the file views.py. Technologies and languages were used, such as HTML, CSS, Javascript and Bootstrap.

The correspondent template file for this application has the name of weather.html.

6.0 UA Newsletter

6.1 The Application



Novo espaço para incentivar comunidade a viver a UA em toda a sua plenitude

Um novo espaço foi inaugurado na chamada zona técnica da Universidade de Aveiro (UA), mais conhecida na comunidade académica por "Catacumbas", traduzindo para um local concreto o que a Reitoria designa como um novo conceito de viver a UA. Ou seja, viver a vida académica em toda a sua plenitude. Com a inauguração, o Reitor Paulo Jorge Ferreira lança um desafio à comunidade para dinamização do novo espaço, "Viver a UA".

Wed, 7 Nov 2018 00:00:00 GMT



UA debate em Bruxelas o futuro do ensino superior

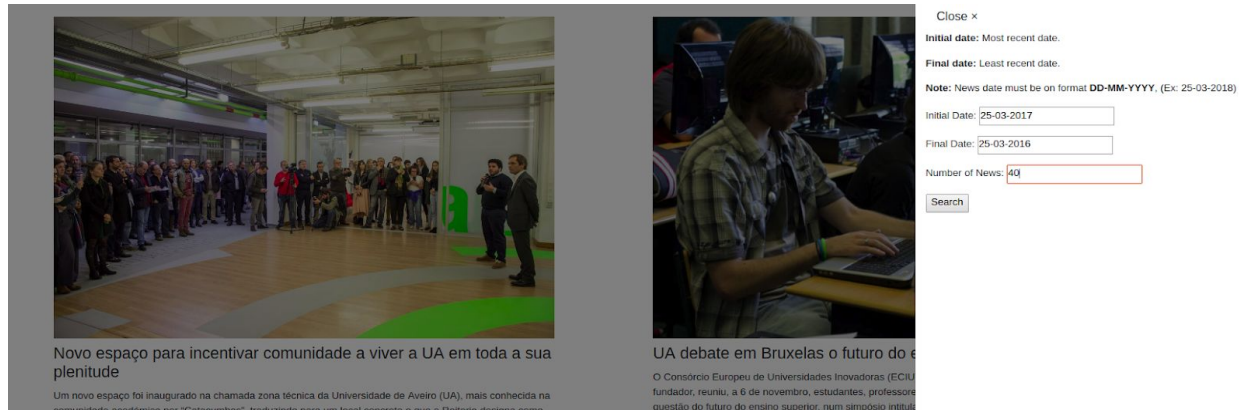
O Consórcio Europeu de Universidades Inovadoras (ECIU), da qual a Universidade de Aveiro é membro fundador, reuniu, a 6 de novembro, estudantes, professores e especialistas, em Bruxelas, para debater a questão do futuro do ensino superior, num simpósio intitulado: "Aprendendo para o amanhã, hoje: universidades do futuro 2040".

Wed, 7 Nov 2018 00:00:00 GMT



On this section the client can see the newsletter of University of Aveiro with the use of the University of Aveiro Newsletter RSS Feed. The user can see the image, title, description and date of the New. Also, if the user wants to know more about the new, it can click on it and it will redirect to the original news provided by University of Aveiro Newsletter site.

Also, if the user clicks on the rightmost bar, the user is able to search news by the recent and latest date. Also it is capable of choose the maximum number of news to be displayed on the web application.



6.2 Technology Used

The information of each News from the University of Aveiro was gather through the following url: https://uaonline.ua.pt/xml/contents_xml.asp?&lid=1&i=11. The data provided is on RSS Format.

The following give us data about each News from the Jornal da UA website.

The class responsible to fetch the data about the University of Aveiro Newsletter is UANews from the file uaservices.py. This data is fetch by the use of the library lxml and validated with the made Schema UANews.xsd. Data can be then structured on a python dictionary format with the purpose of easy manipulation of data to process and display with Django Frameworks.

To fetch specific News we use the url queries concatenated with the respective url service.

For instance:

- If we want to fetch news from the most recent data input, we concatenate:
di=<dd-mm-yyyy>, on the url with a date format (For example, 25-03-2017)
- If we want to fetch news to the most latest date input, we concatenate:
df=<dd-mm-yyyy>, on the url with a date format(For example, 25-03-2016)
- If we want to display n number of news, we concatenate:
n=<n>, on the url, being n an integer.

An example:

If the user asks for the most 40 recent news from 25-03-2016 to 25-03-2017, the UANews class will fetch specifically data from the url:

https://uaonline.ua.pt/xml/contents_xml.asp?n=40&di=25-03-2017&df=25-03-2016

All the concatenation mentioned above with the APIs url is done by the module of "specific_fetch" from the class UANews of the file uaservices.py.

To notice, UANews makes of use of the abstract class XMLService to fetch and validate the obtained xml data.

We used Django Framework to display the data from the file views.py. Technologies and languages were used, such as HTML, CSS, Javascript and Bootstrap.

The correspondent template file for this application has the name of news.html.

6.3 XML Schema

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="rss">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="channel" type="channelT" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="imageType">
    <xs:all>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="url" type="xs:string"/>
      <xs:element name="link" type="xs:string"/>
      <xs:element name="width" type="xs:integer"/>
      <xs:element name="height" type="xs:integer"/>
    </xs:all>
  </xs:complexType>
  <xs:complexType name="itemType">
    <xs:all>
      <xs:element name="guid" type="xs:string"/>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="link" type="xs:string"/>
      <xs:element name="description" type="xs:string"/>
      <xs:element name="pubDate" type="xs:string"/>
    </xs:all>
    <xs:anyAttribute processContents="skip"/>
  </xs:complexType>
  <xs:complexType name="channelT">
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="link" type="xs:string"/>
      <xs:element name="description" type="xs:string"/>
      <xs:element name="language" type="xs:string"/>
      <xs:element name="copyright" type="xs:string"/>
      <xs:element name="lastBuildDate" type="xs:string"/>
      <xs:element name="ttl" type="xs:string"/>
      <xs:element name="image" type="imageType"/>
      <xs:element type="itemType" name="item" maxOccurs="unbounded" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

7.0 SAC (Serviços Academicos)

7.1 The Application

A	Lic. (1º ciclo), Mestrado (2º ciclo) Latest: 137 Line size: 3	D	Atendimento Prioritário Latest: 15 Line size: 0
G	Mobilidade Erasmus (Alunos UA) Latest: 3 Line size: 0	C	Doutoramentos, Agregações Latest: 46 Line size: 0
X	Exchange / Intercâmbio (Incoming) Latest: 5 Line size: 0		

On this section, the User can check the status of each service from *Serviços Acadêmicos e Administrativos* of University of Aveiro. The data that is displayed from each service are: The name and letter of the service, the line size and the last attended number ticket/client.

7.2 Technology Used

The information of each SAC Service was gathered through the following url: <http://services.web.ua.pt/sac/senhas>.

The following gives us data about the services of SAC from the University of Aveiro. The data format is XML. All the data from the this service is fetched by the class `SACService` of the file `uaservices.py`. To notice, the class `SACService` makes of use the abstract class `XMLService` to validate and fetch the data, so as other functionalities. The XML data gathered is validated by a made XML Schema with the name of `stubs/SACServiceSchema.xsd`. After that, all the desired data to process and display on the web application is stored in a dictionary. This is gathered specifically and validated with the python library `lxml`.

We used Django Framework to display the data from the file `views.py`. Technologies or languages were used, such as HTML, CSS, Javascript and Bootstrap.

The correspondent template file for this application has the name of `sac.html`.

7.3 XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="result">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="items" type="itemsT"/>
      </xs:sequence>
      <xs:anyAttribute processContents="skip"/>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name="char">
    <xs:restriction base="xs:string">
      <xs:length value="1" fixed="true"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="itemsT">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element name="item" type="itemT"/>
    </xs:sequence>
    <xs:attribute name="count" type="xs:integer"/>
    <xs:anyAttribute processContents="skip"/>
  </xs:complexType>

  <xs:complexType name="itemT">
    <xs:all>
      <xs:element name="letter" type="char"/>
      <xs:element name="desc" type="xs:string"/>
      <xs:element name="latest" type="xs:integer"/>
      <xs:element name="wc" type="xs:integer"/>
      <xs:any minOccurs="0" maxOccurs="unbounded" processContents="skip"/>
    </xs:all>
    <xs:anyAttribute processContents="skip"/>
  </xs:complexType>
</xs:schema>
```

7.4 Challenges

The Python Library lxml wasn't capable of validating the XML Schema shown above (above requires XSLT3.0) , being tested and validated before with PyCharm tool for XML and XML

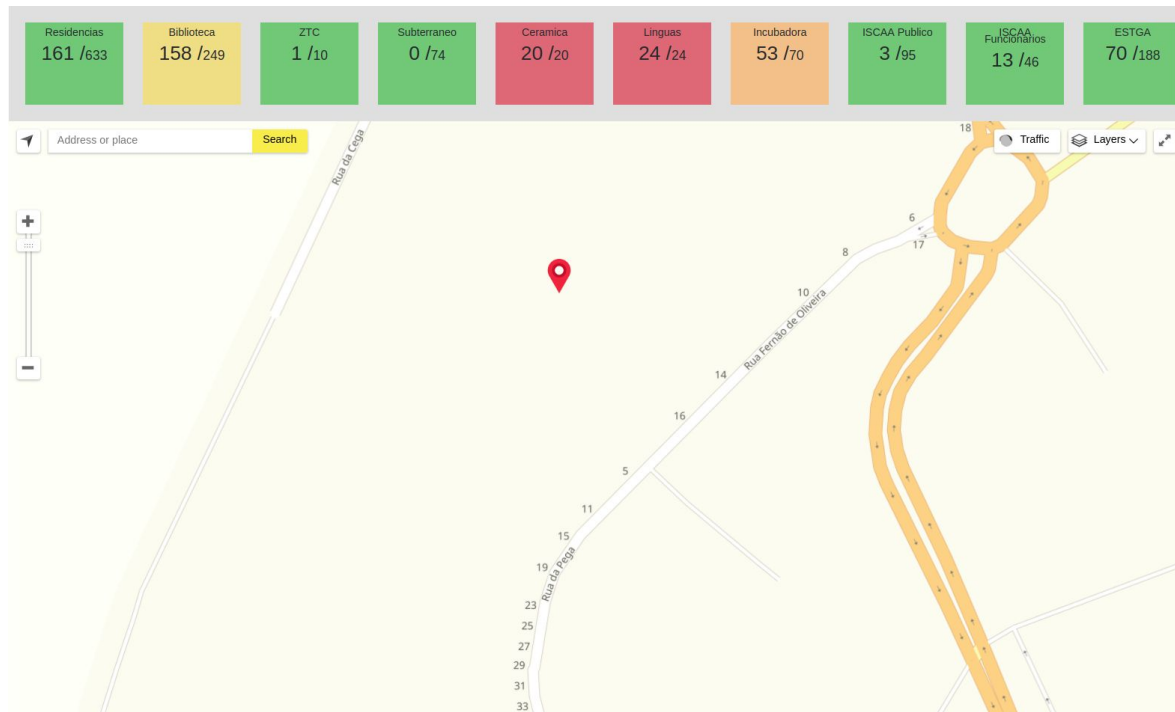
Schema validation. The problem was that the current version of the library lxml is incapable of validating a node “any” inside of a node “all” on a XML Schema, hence not validating the complete XML Schema document and the XML data gathered for this section.

The opted solution was to resort to the Basex Server with custom libraries (basex/) for validation XML data whose schemas has this kind of pattern.

So it is necessary to execute the basex server before using this section.

8.0 Parking Lot

8.1 The Application



On this section the client is able to know which of the Parking Lots of the University are available and those that are not. The client is capable of knowing the Parking Lot location by clicking the respective Parking Lot Square. Each square represents the capacity of the parking lot and the number of lots already occupied. Each square changes its color with the current status of the parking lot. If most of the lots are available the square color will become green, otherwise the color will become red.

8.2 Technology Used

The information of each parking lot was gathered through the following url: <http://services.web.ua.pt/parques/parques>. The following gives us data about the Parking Lots of the University. The data format is not XML, but JSON format. Because this format does not contemplate the objective of this project, it was decided to parse the data from JSON to XML. This is done on the class UANews of the file uaservices.py. To notice, the class UANews makes use of the abstract class XMLService to validate, fetch the data and other functionalities. Inside the class UANews data that is fetched is translated to a desired XML Format, in the module JSON2XML, and then it enters to a process of validation by a made XML Schema with the name of UAParkingSchema.xsd. After that, all the desired data to process and display on the web application is stored in a dictionary. This is gathered specifically and validated with the python library lxml.

We used Django Framework to display the data from the file views.py, where it also processes data to determine if a correspondent number is less than zero or to determine which status color will be given following the current status of the parking lot, for example.

For the display of the map it was used Yandex Maps API. Other technologies or languages were used, such as HTML, CSS and Javascript.

The correspondent template file for this application has the name of parkinglot.html.

8.3 XMLSchema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="Estacionamentos" type="estacionamentosT">

  </xs:element>

  <xs:complexType name="estacionamentosT">
    <xs:sequence >
      <xs:element name="Timestamp" type="xs:integer"/>
      <xs:element name="Estacionamento" type="estacionamentoT" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute processContents="skip"/>
  </xs:complexType>

  <xs:complexType name="estacionamentoT">
    <xs:all>
      <xs:element name="ID" type="xs:string"/>
      <xs:element name="Nome" type="xs:string"/>
      <xs:element name="Latitude" type="xs:float"/>
      <xs:element name="Longitude" type="xs:float"/>
      <xs:element name="Capacidade" type="xs:integer"/>
      <xs:element name="Ocupado" type="xs:integer"/>
      <xs:element name="Livre" type="xs:integer"/>
    </xs:all>
    <xs:anyAttribute processContents="skip"/>
  </xs:complexType>
</xs:schema>
```

8.4 Challenges

As it was mentioned, the data provided by Academic Playground & Innovation is on JSON Format and not in XML. So parsing this data from a format to another, respectively, was needed to contemplate the objectives of this Project.

Also, the data provided is not calibrated, some values exceed the total capacity of lots and some of them are negative values. For the last situation it was decided that when this happens, the web application will display zero instead of a negative value.



9.0 Conclusion

In this project we were able to develop a useful application employing XML technologies and Django framework.

The overall result of our solution was positive as we managed to make an application that is useful to us and the academic community, pleasant to use and with a good level of exploration of the mentioned technologies.

Django simplified many aspects of the development that would have otherwise made it very difficult to focus on the essential part of this project.

We were able to understand the reality of semi-structured data and how to properly handle and manipulate it.

10.0 Installation and Execution

10.1 Python dependencies

This project depends on the packages Django, lxml and BaseXClient:
\$ pip install Django lxml BaseXClient

10.2 Execution

1. Enabling basex server (ex: ./basex/bin/basexserver)
2. Enabling django (ex: python manage.py runserver)