Taraxacum
Design Description #1

Version 1.1

Revision History

Date	Version	Description	Author
2011-10-23	0.01	Initial Draft, Introduction, System specification	Magdalena Juric
2011-10-23	0.02	Class diagram picture	Tomislav Bronic
2011-10-23	0.03	System architecture pictures	Mateo Klarin Petrina
2011-10-26	0.04	Error handling, Client server architecture,	Magdalena Juric
2011-10-26	0.05	MVC pattern, Project Scope, System architecture	Mateo Klarin Petrina
2011-10-26	0.06	Class diagram text	Tomislav Bronic
2011-10-26	0.07	Some changes in Conceptual design	Magdalena Juric
2011-10-27	0.08	External interfaces	Anne Jon Schoonhoven
2011-10-27	0.09	Class diagram text and picture change	Tomislav Bronic
2011-10-28	1.0	Added database model, Finalized document	Magdalena Juric
2011-11-27	1.1	First iteration update, changed detailed design section.	Magdalena Juric

Table of Contents

1. Introduction

1.1 Purpose of this document

The purpose of this document is to describe the system design of the Taraxacum project, to describe the architecture and to give a broad prospective of the design of the entire system. This project is part of the Distributed Software Development course held in 2011. This course is joint course between Mälardalen University (MDH) in Sweden and University of Zagreb (FER) in Croatia.

1.2 Intended Audience

The Intended Audiences of this document are:

- Team members to use it as a reference during the Implementation phase of the project.
- Customer to understand the project scope.
- Team supervisor to have an overview of the project work.
- Future developers of the system, who should know about the basics of the system before they start upgrading it.

1.3 Scope

This document describes high and low system architecture of Taraxacum project.

1.4 Definitions and acronyms

1.4.1 Definitions

Keyword	Definitions
Taraxacum	The project's title

1.4.2 Acronyms and abbreviations

Acronym or abbreviation	Definitions
MDH	Mälardalen University
FER	University of Zagreb
MVC	Model View Controller
CSS	Cascading Style Sheets
GUI	Graphical User Interface
HTML	Hypertext Markup Language
AJAX	Asynchronous Javascript and XML
MSSQL	Microsoft SQL Server
HTTP	HyperText Transfer Protocol

1.5 References

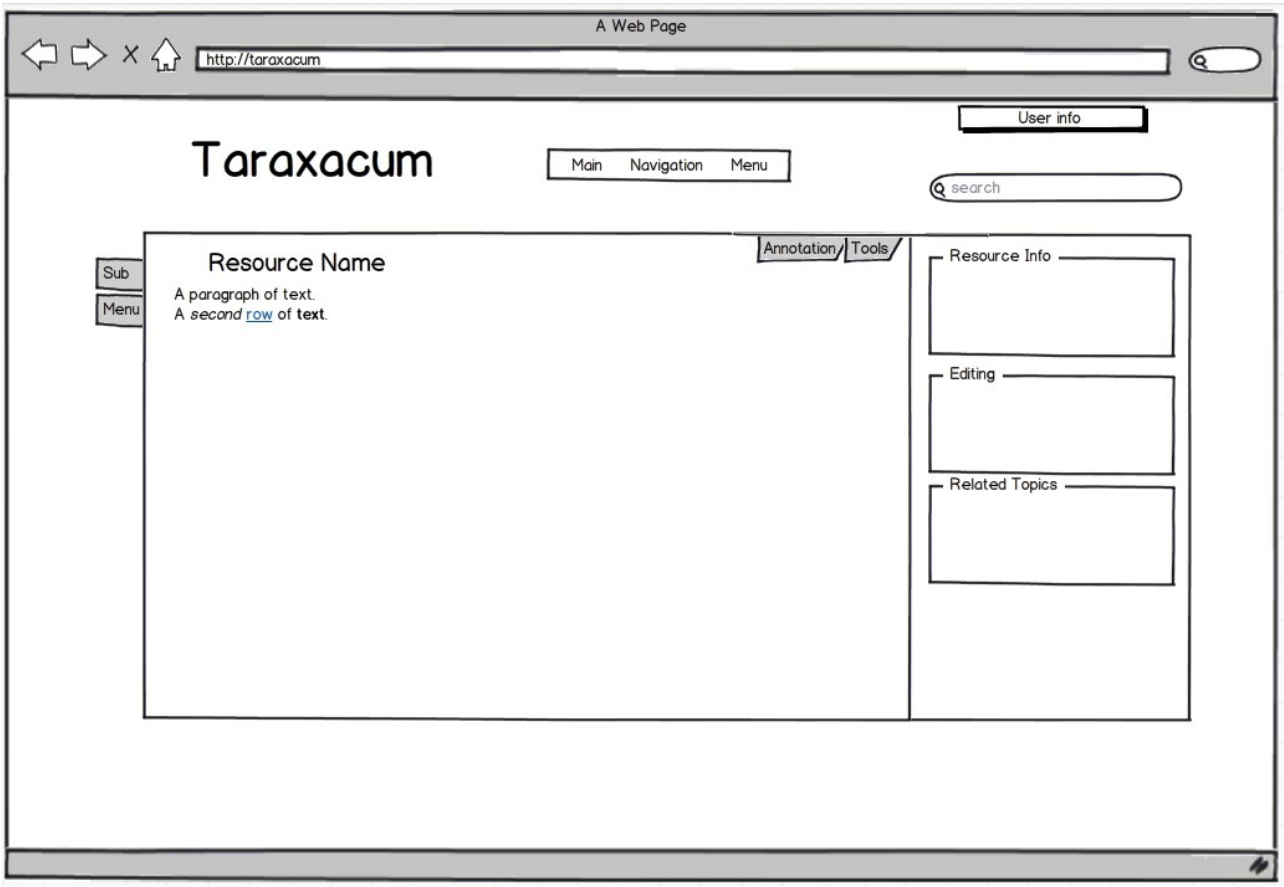
Project proposal: http://www.fer.unizg.hr/_download/repository/seve-taraxacum.pdf
Project Homepage: http://www.fer.unizg.hr/rasip/dsd/projects/taraxacum

2. External interfaces

2.1 Web Application

The Web application provides the users of the system an interface to interact with the system. The system is intended to be used by users of two types, teachers and students.

2.1.1 GUI description



In our initial version of the Graphical User Interface we modeled the interface around the core element of the application, the resource. The layout is furthermore widely formatted, which gives us the flexibility of adding or rearranging certain elements when new requirements will rise.

On the top, one can see the main navigation of the web application. This menu will stay consistent over all the pages. On the left, one can see a context related sub menu. This menu will be used to perform related actions to the resource, such as viewing its exercises, or its comments.

On the right side one can find a column containing Meta information about the current viewing item, and optional administrative tasks. Furthermore, about the resource there is room for annotation tools. Information and actions concerning the user are showed in the upper right corner.

2.2 The core of the system

The core performs all the logical operations, handles and manipulates the data that flows within the system. Hence being the brain of the system all communication happens through it at some point of time.

2.3 Web service (optional)

Web service is used like interface for mobile applications. It will be connected to core of the system.

2.4 Mobile Application (optional)

The mobile application will be developed for interacting with the system when on the move.

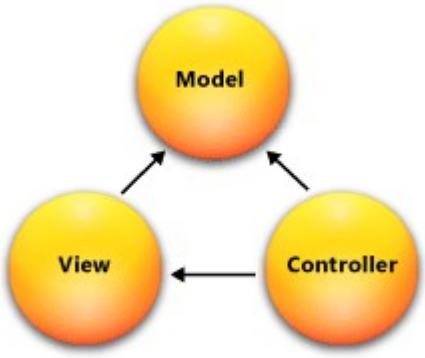
3. Software architecture

3.1 Conceptual design

3.1.1 Client – server architecture

The Taraxacum system is a web based system using standard client-server architecture. On the client side, client computer uses web browser to access the system. Standard HTTP protocol is used in the client-server communication. On the server side, web server receives requests from the client, handles them and sends a response to the client. Web server communicates with the database, to load and save data.

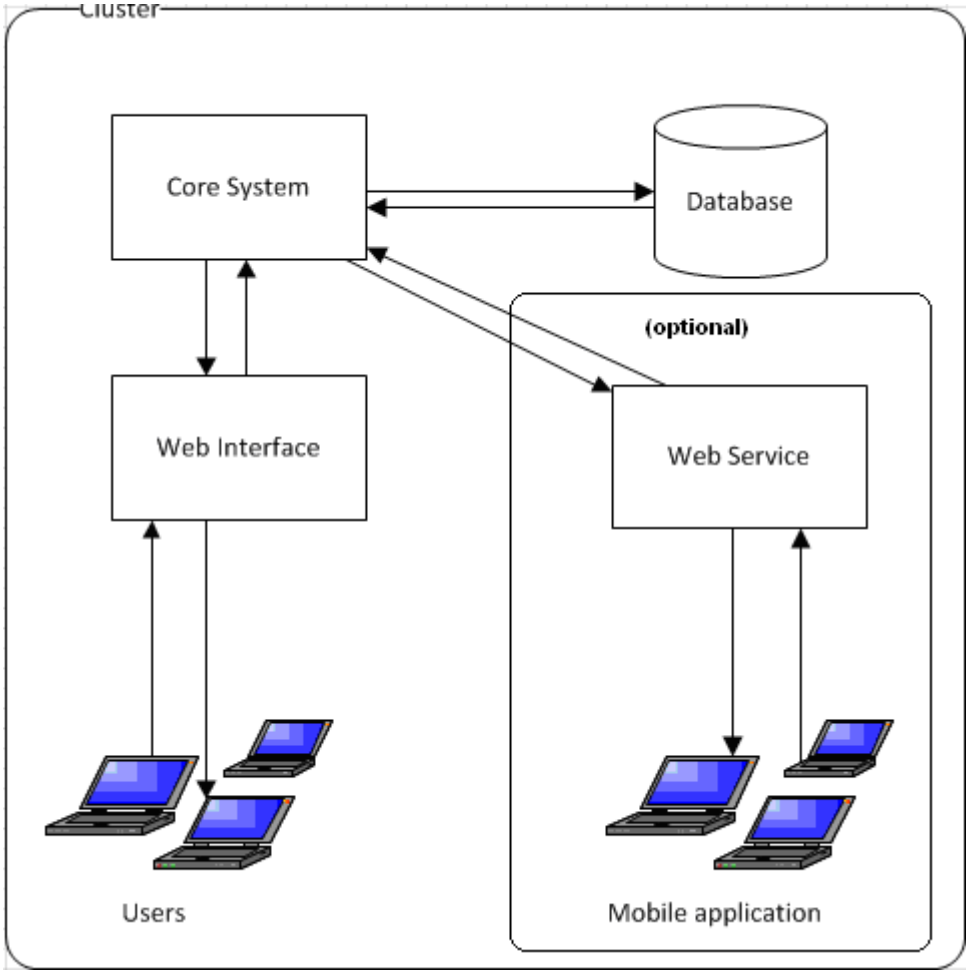
3.1.2 The MVC pattern



Model-View-Controller(MVC) architectural pattern that separates an application into three main components: the model, the view and the controller. The pattern isolates "domain logic" (the application logic for the user) from the user interface (input and presentation), permitting independent development, testing and maintenance of each (separation of concerns).

3.1.3 Project scope

The end product of this project will be fully functional web application. Fully functional implies that are requirements specified in document Requirements design will be implemented. Additionally, if time frame permits, web service and mobile application will be developed.



Project scope diagram

3.1.4 System architecture layers

Our system architecture has more layers:

- Database Layer
- Data Access Layer
- Model
- Controller
- View

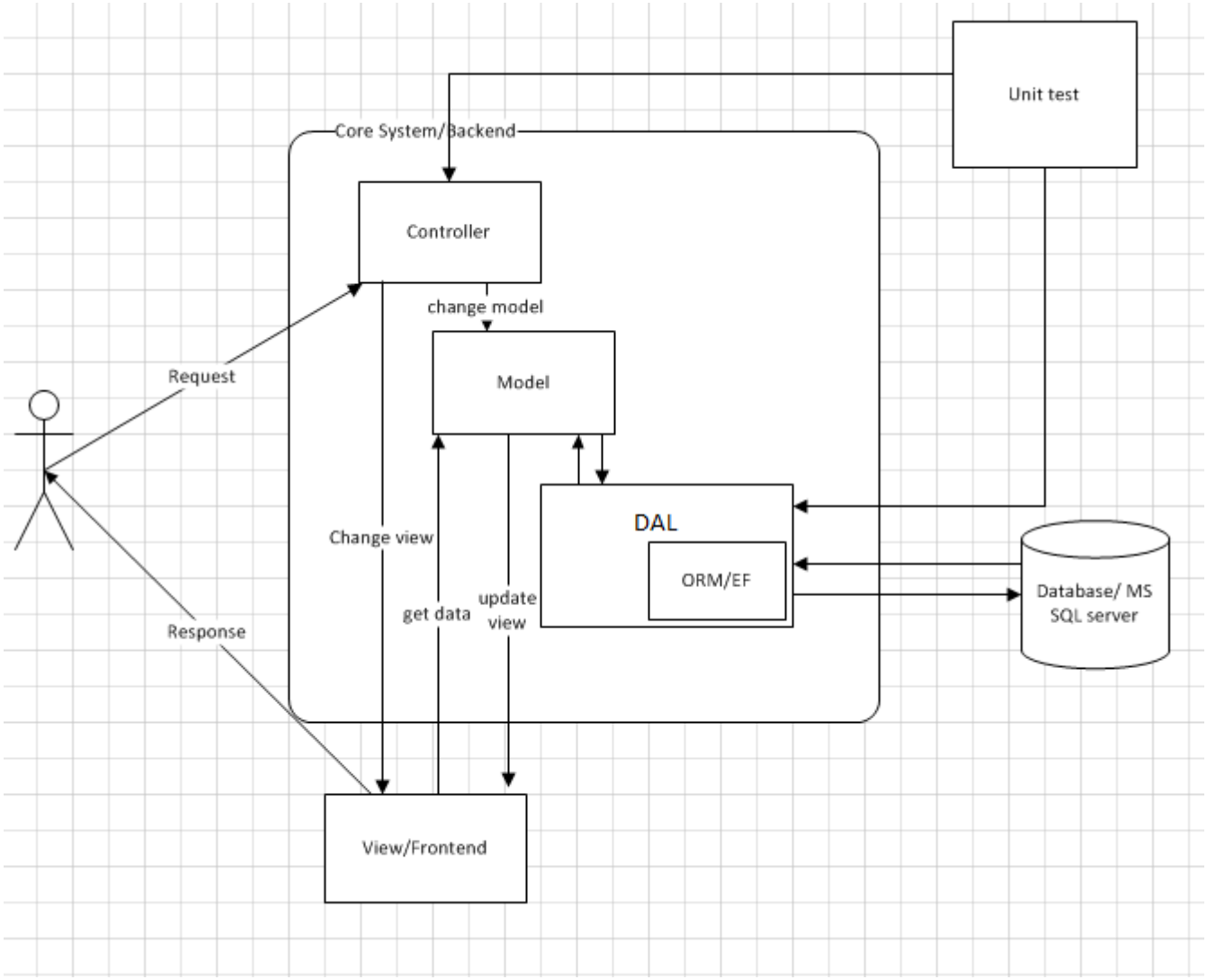
Database layer contains MSSQL database which is running on Microsoft SQL server.

Data Access layer (DAL) contains object relational mapping files. Through this layer a model maps to a database tables with the entries in the table representing the state of the application.

Model represents domain logic and state of particular aspect of the application.

Controller handles interactions and updates the model to reflect a change in state of the application, and then passes information to the view.

View accepts necessary information from the controller and renders a user interface to display that.



System architecture diagram

Backend of application handles user request, data processing and communication with database. Backend is subject to intensive testing throughout development phase.
Database permanently stores and retrieves data.
Frontend displays data to the users.

3.2 System specification

The Taraxacum system is powered by a web server hosted on a virtual machine and available on the address. The MSSQL database management system that is used on this project is hosted on the same virtual machine.

Various technologies and tools are going to be used on this project:

- Frontend
  - ASP.NET
  - HTML
  - CSS

  - jQuery
  - AJAX
- Backend
  - C#
  - ORM - Entity framework
- Database
  - MS SQL

3.3 Error handling

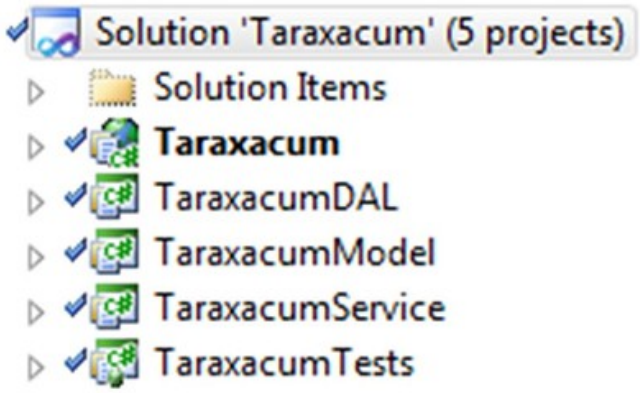
Error	Action
User input error	Cancel the current input. Notify the user.
Data access error	Notify the user.
Server error	Notify the user.
Client error	Notify the user.

4. Detailed software design

4.1 Code division

Code is divided into five projects.

- Taraxacum is used for frontend.
- TaraxacumDAL is used for DAL code.
- TaracacumModel is used for model classes.
- TaraxacumService is for service classes.
- TaraxacumTests is for unit tests.



4.2 Class diagram – Model layer

The main entities in the system are **Resource**, **Topic**, **Course** and **User**.

A resource can be multiple things: an external link, a file or text. The idea is to have a repository of all resources registered / created in the system. Each resource can belong to more courses and has one topic.
With the hierarchy composition it's easy to add a resource to a course, or add resource to topic of a course.

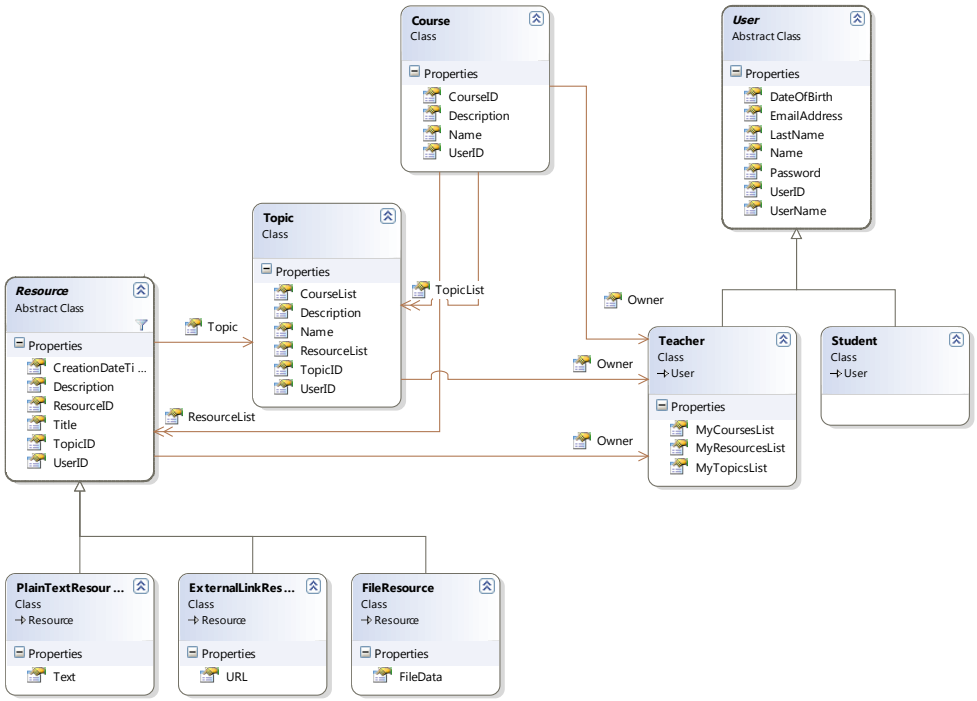
Topic represents another type of classification of resources. Topic represents the subject of a resource (for ex. c#, java, UML, etc.). One topic can contain more different resources. Topic can belong to many courses.

Course represents collection of resources and topics of that specific course. Teacher can add his own resources and topics to it, or can add materials from existing topics or resources to his course.

Resources, topics, courses and users will have their own repository in DAL layer, and therefore can be searched and found individually.
The main idea in the system is to be able to search for content per resources, topics or courses.

User represents type which can be student or teacher. They have different view on the system depending on which role they are in.

For the Alpha version of the system, a less-detailed class diagram was created, which includes only the core-functionality classes.



Iteration 1 - Core Class diagram

Diagram for 2nd iteration will be extended **Exercise**, **Flashcard** and **Note** functionalities.

Diagram for 3rd iteration will be extended with **Comment** and **Rating** functionalities.

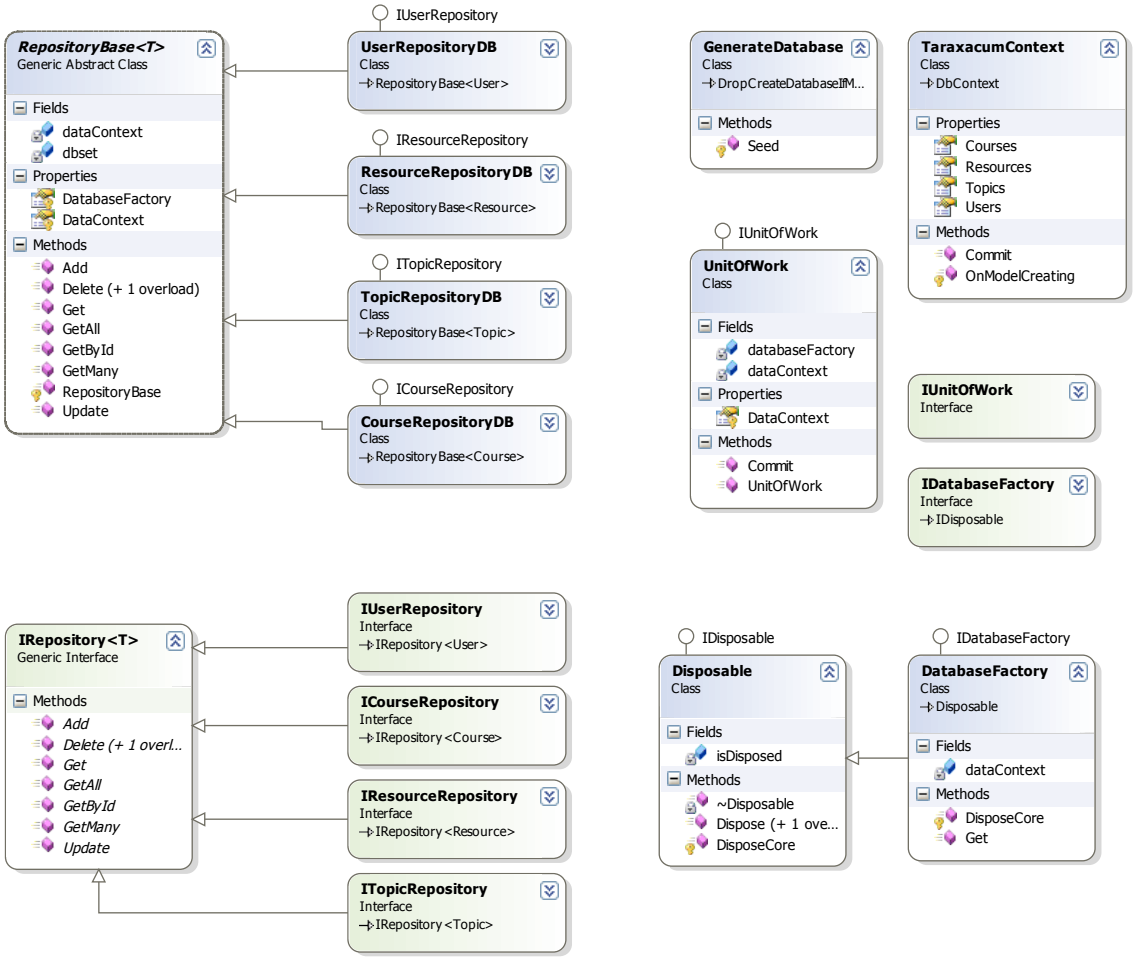
4.3 Class diagram – Data access layer

Data access layer encapsulates all functions for accessing to database and provides interfaces to upper layers of the system. It consists of Infrastructure part, Repositories and classes for Database generating.

In Infrastructure, there are defined generic interfaces and generic classes for working with databases and sessions. Those are Database Factory, Repository Base, Unit of Work and Disposable.

In Repositories, there are defined repositories and interfaces for every specific entity from the model. They provide CRUD access through their functions.

Generate Database class contains seed function which seeds data to database using Entity framework. In Taraxacum Context additional mappings for model are defined.

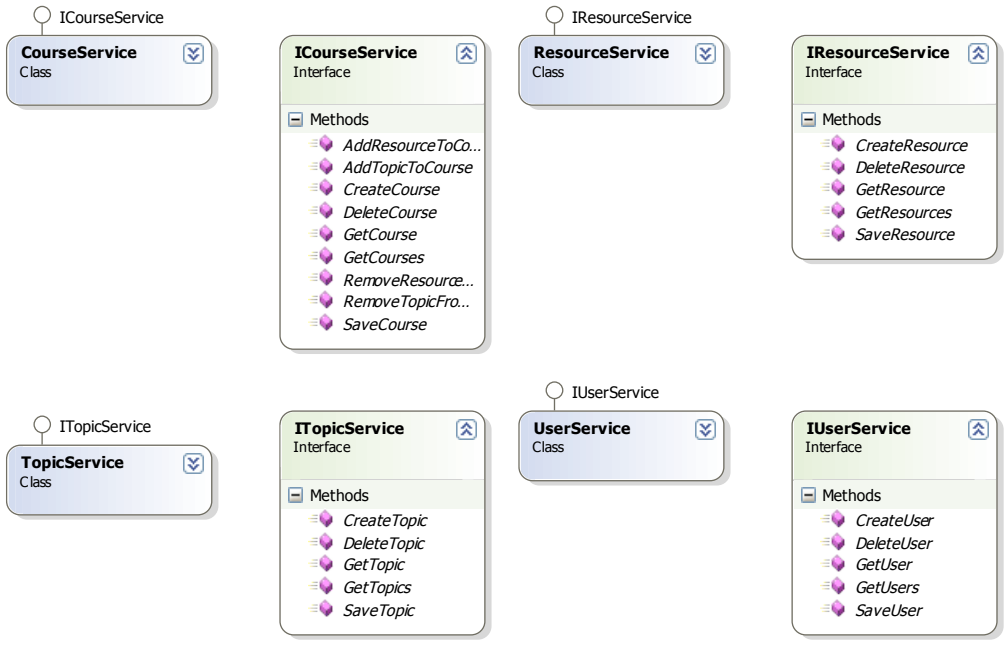


Iteration 1 - DAL Class diagram

4.4 Class diagram – Service layer

Service layer is middle layer between DAL and frontend layers. It provides interface and takes care of same context for every query.

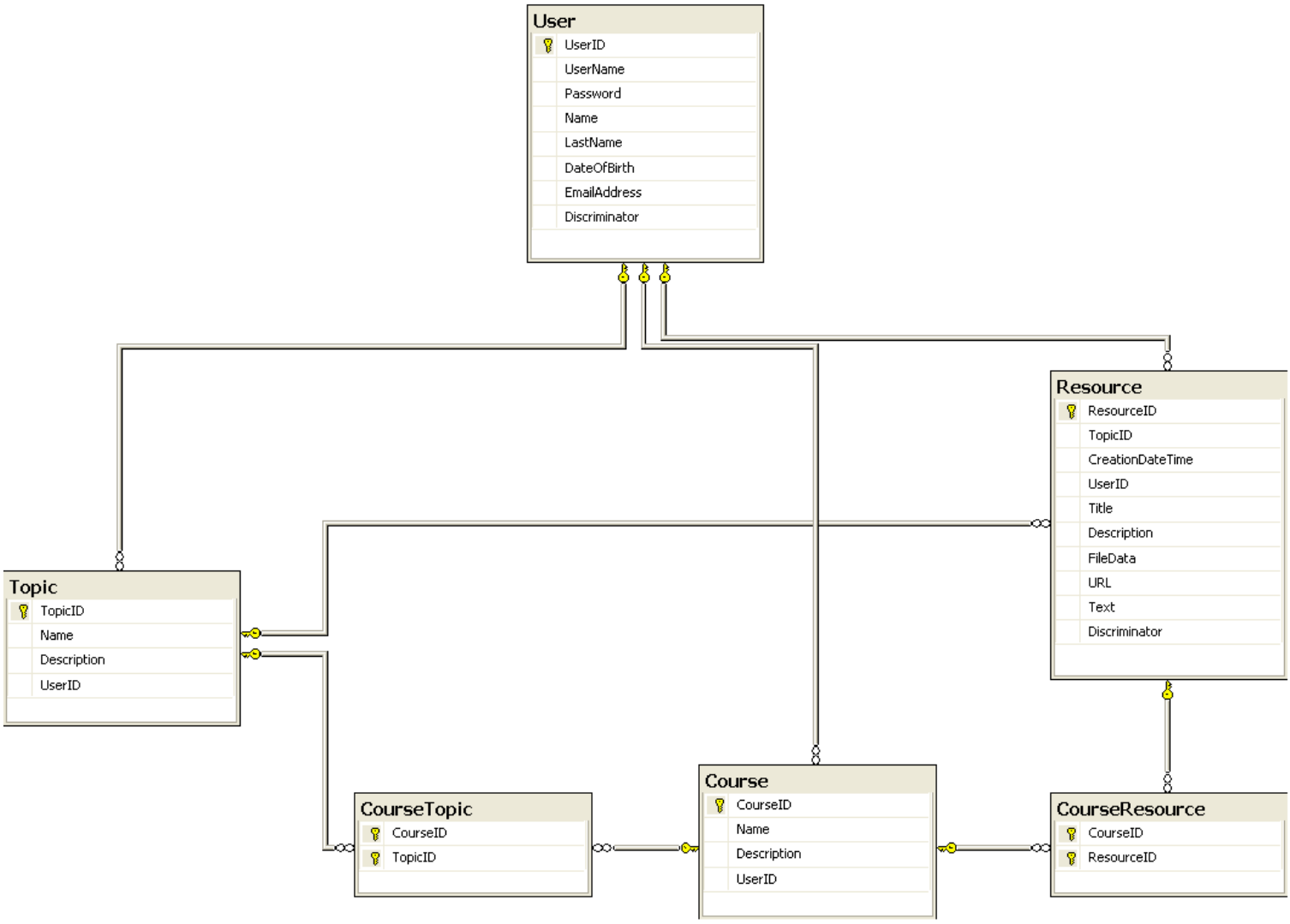
Every entity from the model has its own service class. They support methods which are needed in upper layers.



Iteration 1 – Service Class diagra

4.5 Database model

This picture shows relational database model, with tables and their connections.
All classes are mapped to data tables using Entity Framework mappings.



Iteration #1 - Database model diagram

5. Approvals

Name	Title	Date yyyy-mm-dd	Signature