

# Central Connecticut State University

**University Student Portal using the *Semantic Web Approach***

CS-595 Capstone Final Report

**Team Members:**

*Nagaharika Ogirala*

*Swetaa Ganesan*

*Sherin Cheriyan*

**Advisor:**

*Dr. Neli Zlatareva*

**Spring 2021**

**Table of contents:**

1. **Introduction**
   1. **Domain**

1.1.A Existing System

1.1.B Proposed System

1. **Knowledge Gathering**
   1. **Domain – Department of CS**
   2. **Skillset**

2.2.A Semantic Web

2.2.B Spring Boot Micro-services

2.2.C Angular

1. **Design and Analysis**
   1. **Technical Stack Decisions**
   2. **Semantic Web Schema Designs**
   3. **It all fits (Integration)**
2. **Planning and Development Strategies**
   1. **Agile Development**
3. **Implementation**
   1. **Control Flow**
   2. **Services and Endpoints Used**
4. **Application**
5. **Possible Extensions**
6. **Conclusion**
7. **References**

**1.Introduction**

The university student portal is developed using the Angular User Interface and back-end is developed with Java using Spring Boot, in which the data is created and updated through the Ontology model developed using Protégé. The ontology creates a relationship between various entities and classes, it also describes how these entities can be grouped according to their similarities and differences. In our ontology model, we are creating two different ontologies: Faculty Ontology and Courses Ontology. The Faculty Ontology contains details of the faculty and staff members. The Course Ontology contains the course details, time, days and the semester, in which the courses are provided by the department for both graduate and under-graduate students. These ontologies are then combined, thus providing a relationship between the faculty and the course, where we can then obtain the details of the professors teaching the different courses.

Jena API is used to maintain the schema of the ontology using various Inference and Ontology models. This application is built in Java SE 10 using the Eclipse IDE. The Angular framework is used to build the User Interface for the data available in the Ontology models. Spring Boot works at the middle layer, wiring the Angular User Interface with the ontology models and/or database for student login.

The intent of the web application is to help students log into their University account, as well as register for courses and view their registration status. We are using two possible data-sources, one being Protégé to populate domain information(turtle files) and Database to store the login information and student registration status. Jena API is used to integrate the models in runtime.

* 1. **Skillset**

**2.2.1 Semantic Web**

The Semantic Web is a vision about an extending the existing World Wide Web. The data in www is unstructured if this data is made a part of the new Semantic web, can be referenced directly by their unique identifier, called Universal Resource Identifier (URI). URIs are interconnected in a graph infrastructure thus comprising a huge library of information that can be easily and uniformly accessed by Semantic web applications.

The benefit of using Semantic web over the existing web is, the Semantic web facilitates the machines to understand the information allowing the users to perform task effortlessly. To enable the encoding of semantic with the data, technologies such as Resource description framework (RDF) and  Web ontology Language(OWL) are used. These technologies are used to formally represent metadata.

**2.2.2 Spring Boot Micro-services**

Microservices are increasingly used to create larger, more complex applications that are better developed and managed as a combination of smaller services that work cohesively together for larger, application-wide functionality. Spring Boot is a java-based framework which builds a standalone environment which communicates with many microservices available in our application.

The benefits of using Microservice architecture is it would allow the developers to scale up or down the size of the project independently. This feature is also useful in decreasing the expenses for the overall application also the failure of one module would not affect the whole operation of the application.

**2.2.3 Angular Framework**

Angular is a component-based framework for building scalable web application. The basic building blocks of the Angular framework are Angular components that are organized into NgModules. NgModules collect related code into functional sets, an Angular application is defined by a set of NgModules. An application always has at least a root module that enables bootstrapping, and typically has many more feature modules.

Angular applications are built using TypeScript language, a superscript for JavaScript, which ensures higher security as it supports types (primitives, interfaces, etc.). It helps catch and eliminate errors early when writing the code or performing maintenance tasks. With Angular, we do not need any additional getter and setter functions. Since, every object it uses is POJO, which enables object manipulation by providing all the conventional JavaScript functionalities.

**3. Design and Analysis**

**3.1 Technical Stack Design**

**3.1.1 Semantic web Technologies:**

The data in www is unstructured if this data is made a part of the new Semantic web, can be referenced directly by their unique identifier, called Universal Resource Identifier (URI). URIs are interconnected in a graph infrastructure thus comprising a huge library of information that can be easily and uniformly accessed by Semantic web applications.

**Turtle:**

A Turtle file allows writing down an RDF graph in a compact textual form. An RDF graph represents information using semantic [triples](http://www.w3.org/TR/rdf11-concepts/#dfn-rdf-triple) consisting of a subject, predicate and object. Each item in the triple is expressed as a Web URL. Turtle provides a way to group three URIs to make a triple, and ways to abbreviate such information. Subjects are referenced by a number of predicates.

**Protégé 5.5.0**

Protégé is an editor which helps us to create ontology models. The Protégé version we are using is Protégé 5.5.0.

**Apache Jena – API 3.15.0**

**Openllet 2.6.5**

**3.1.2 Java Technologies:**

**Java SE 10**

**Apache Maven 3.6.3**

**Spring Boot**

**Bootstrap 3.3.7**

**3.1.3 Angular**