POLYTECHNIQUE
MONTRÉAL

LE GÉNIE
EN PREMIÈRE CLASSE

Seodin

Guide: Dr. Fabio Petrillo

Professor: Prof. Yann-Gaël Guéhéneuc

Github: https://github.com/ptidej/seodin

Project Report by:

Dipti Ranjan Sahu

IIT Bombay

Contents

Abstract

This project aims at developing an open data web application to manage a library of software engineering studies. The web application will have access to a database containing large number of software studies, which can be used for study or research purposes. A study will contain different types of artifacts, arranged in different subdirectories. A study contains different interviews and think-alouds (audios, videos and notes), contributed by several developers. The project will also contain the source-code, test-cases, defects and interaction-logs of the studies. One can easily access the existing studies, or can contribute to a new or existing study by uploading his/her work. This web application uses spring boot web security configuration, which provides the HTTP basic security for all end points. The graphical user interface (GUI) of the application is quite simple and user-friendly.

Introduction

Motivation:

The Ptidej team develops theories, methods, and tools, to understand, evaluate, and improve the quality of software systems by promoting the use of idioms, design patterns, and architectural patterns. It wants to formalize patterns, to identify occurrences of patterns, and to improve the identified occurrences. It also wants to evaluate experimentally the impact of patterns on the quality of software systems. This project leads us to connecting various studies that are available in the software engineering community, and identifying patterns common to them, thus making them more useful.

Project Objective:

The primary objective of the project is to provide users access to numerous studies by different developers. Firstly, the web application must have the required security feature, not to be mishandled by anonymous users. The web application should provide the API for handling the application software. It will help in consuming REST API, i.e. it can handle create, retrieve, update and delete (CRUD) of the entities. It will also allow searching artifacts by name, developer or study-title. There will also be audio streaming player to listen to audio files and video streaming player to watch videos online. It will prevent users from downloading audio and video files.

Project Design

Model Class Diagram:

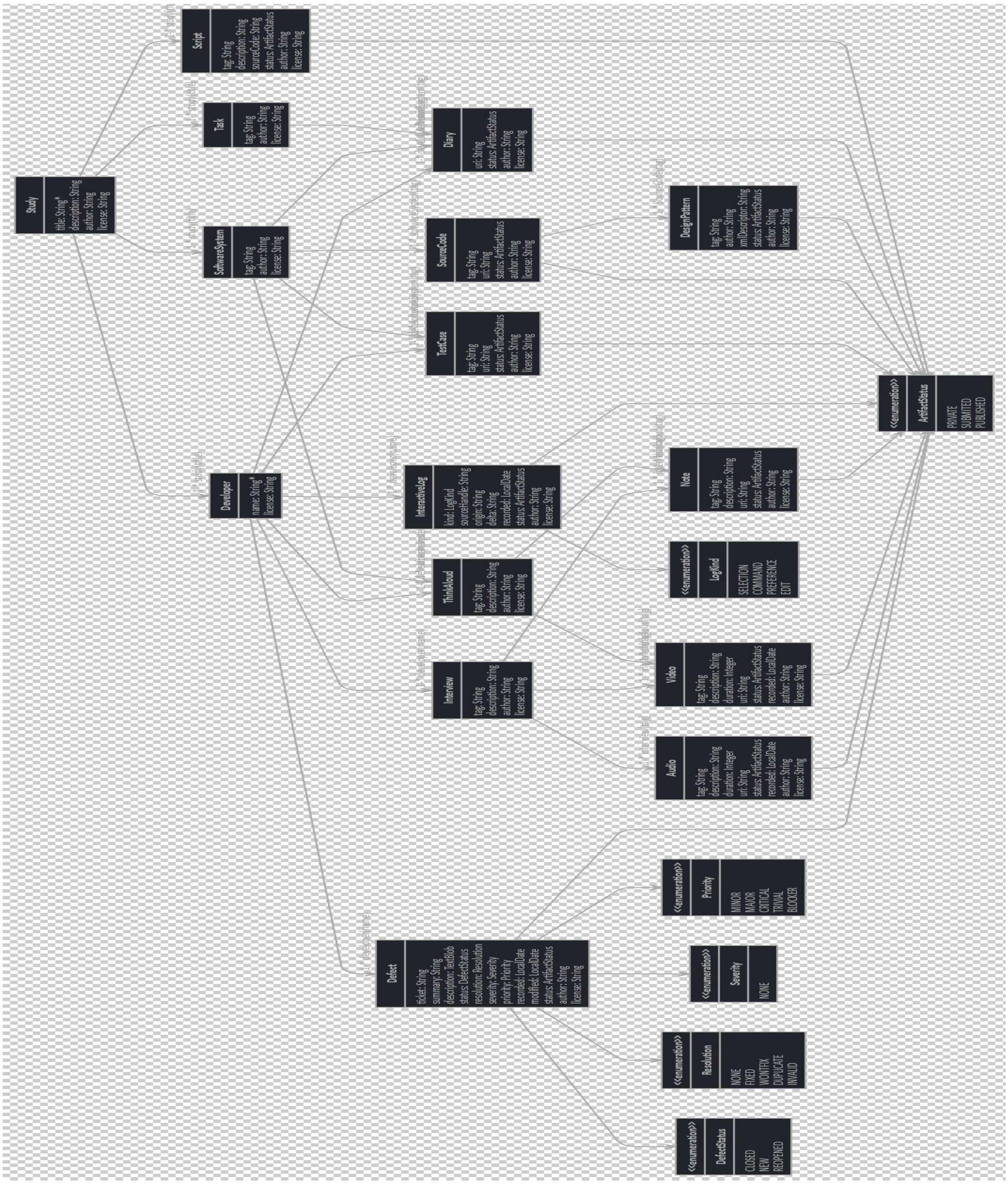

Figure 1. Overall system

A researcher/user can access any study. A study is contributed by different developers. The study is also linked to software systems, tasks and scripts. There are also other entities such as interviews, think-alouds, audios, notes, videos, source codes, test cases, defects, issues, interaction logs, diaries, design patterns. All these items are considered as artifacts and interlinked with each other as seen in the Figure 1. These can be searched by study-title, developer or the artifact tag.

Relationships between different entities (One to Many):

- Study to Developers
- Study to Software Systems
- Study to Scripts
- Study to Tasks
- Software System to Source Codes
- Software System to Diaries
- Task to Diaries
- Software System to Test Cases
- Software System to Think Alouds
- Source Code to Design Patterns
- Developer to Interviews
- Developer to Diaries
- Developer to Think Alouds
- Developer to Defects
- Developer to Test Cases
- Developer to Interactive Logs
- Interview to Audios
- Interview to Videos
- Interview to Notes
- Think Aloud to Notes
- Think Aloud to Videos

Project Implementation

Completed Project Subsystems:

- Primary framework including the authentication system
- REST API
- Export Operation
- Video Streaming Player
- Search Module

Primary Framework:

This application is generated using JHipster 4.5.3, you can find the documentation and help at http://www.jhipster.tech/v2-documentation/. JHipster provides tools to generate a project with a Java stack on the server side (using Spring Boot) and a responsive Web front-end on the client side (with Angular and Bootstrap). It can also create microservice stack with support for Netflix OSS, Docker and Kubernetes. The application uses Spring Boot Web Security Configuration to provide HTTP basic security for all end points. It also provides an authentication manager bean with in-memory store and a single user, an application event publisher to publish successful or unsuccessful authentication and denied access. It ignores insecure paths for common static resource locations. It gives different privileges to users and administrators. Users have the right to access the studies, contribute to them, while administrators have the right to change the architecture of the project or add more functions to the applications.

REST API:

The application also provides and uses an API, which allows to create, retrieve, update and delete (CRUD) of the entities. Different categories for the entities are defined using JHipster generator. The dependencies and the relationships between different categories are later defined using the JHipster generator similarly. For details of how to create entities and define relationships, refer the manual. The administrative panel gives the right to make changes in the attributes of the entities, or change the existing data itself.

Export Operation:

We created an export function for downloading a study, organized into subdirectories. The study to be exported is identified by matching its ID against the database. The study is wrapped as a JSON object. Developers, software systems, tasks and scripts related to this study are identified by the study title in the respective repositories. Similarly, entities related to those developers, software systems, tasks and scripts are identified by the following tags in different repositories. The study JSON object contained developers, software systems, tasks and scripts that were also further divided into subcategories. Hence, all the entities related to that study gets wrapped as a JSON object in the hierarchical way as shown in the class diagram. This function downloads a study on a standalone computer, which can be used for further detail analysis and research activities.

Video Streaming Player:

We created a simple video streaming player using flowplayer and real time messaging protocol (RTMP) plugin. Real-Time Messaging Protocol was initially a proprietary protocol developed by Macromedia for streaming audio, video and data over the Internet, between a Flash player and a server. The video player streamed a video from a known server using real time messaging protocol. We tried different types of players and different types of plugin and finally we came to the conclusion that RTMP plugin is the best for streaming videos. It is a basic video stream player and needs to be integrated with our main application.

Search Module:

For easiness in finding a particular study, we built a search module that uses elastic search for finding the study by their titles, and displaying the required results in the table. We also worked on the graphical presentation of the selected study. On clicking the particular study, a table of interviews and think alouds related to that study expand. Furthermore, when an interview is clicked, a popup dialog appears that contains three subsections, i.e. videos, audios and notes related to that interview. Similarly, when a think aloud is clicked, a dialog box for audios and notes pops out.

Performance Evaluation

Testing Strategy:

The web application was checked after every sprint, i.e. around every alternate Fridays. The development of the application was followed by an agile process. At every meeting, some new work was being assigned and some new changes to the existing application was proposed, with focus on process adaptability and user satisfaction. The new changes and additions were implemented and represented in the next sprint. The web application was run on other computers using internal server and against a huge amount of data in database.

Testing Results:

There were several computers involved, in development of the web application. The application was hosted by one of the computers. The other computers were used to test the speed and effectiveness of the application. It began with a slow start of loading for first time on the other computers. After being loaded once, the web application was smooth and robust, and exhibited no major faults. It worked quite well with the large amount of data.

Problem Faced:

- Video streaming player was difficult to implement, and further hard to integrate it in the main application.
- We lacked server for uploading our videos, and then streaming it from the server.
- It took some time around 2-3 sec for opening the application for the first time with the huge database.

Conclusion

JHipster is a new technology and it combines lots of good things and provides many benefit. It brings out the Spring Boot security, Swagger API, AngularJS, HTML5, CSS and lots more. It is a good concept to learn and I am sure it will be helpful in the future.

The web application has lots of utilities and benefits. It will bring a whole lot of software engineering projects accessible by a user, who can use for his own studies or research work. The project is more than half complete and will be full-fledged in short time and can help in bringing the world one step closer to internet of things.

Future Work

- The import function was not built. It is needed to be done using some code because we would be importing a large amount of data, which is not possible manually.
- Video and audio streaming player needs to be implemented in the main app.
- The application needs a good graphical interface to make it look attractive.

References

- http://www.jhipster.tech/
- https://spring.io/blog/2015/02/10/introducing-jhipster
- https://www.tutorialspoint.com/restful/restful_introduction.htm
- http://www.jhipster.tech/api-gateway/
- http://www.jhipster.tech/creating-an-entity/
- http://www.jhipster.tech/jdl-studio/
- http://www.jhipster.tech/using-dtos/
- https://flowplayer.com/docs/setup.html
- http://flash.flowplayer.org/plugins/streaming/pseudostreaming.html
- http://flash.flowplayer.org/plugins/streaming/rtmp.html
- http://www.jhipster.tech/using-elasticsearch/
- https://www.w3schools.com/howto/howto_css_modals.asp