Project Scope and Outline

Natural Deduction Checker

|  |  |
| --- | --- |
| Student Name: | Lee Paterson |
| Student Number: | 201446274 |
| Supervisor Name: | David Bevan |
| Project Name: | Natural Deduction Checker |

**Project Overview**

The aim of this project is to provide a web-based application for checking natural deduction proofs for first-year students taking the class CS103: Machines, Languages and Computation. The application will be used as a teaching aid to the first-year students as it will give feedback based on their input and own deduction, keeping track of mistakes and giving feedback explaining their errors in a clear and easy-to-understand manner.

The application will allow a student to enter the logic formula that they wish to conduct a proof upon. The natural deduction checker will be able to prove the student’s formula, assuming the logic entered is syntactically correct (input will be validated by the app) and if the formula is provable. The student will then begin to input which rules they should apply to the formula, which may be applied to partially completed proofs, not just from scratch. The application will compare the student’s proof with the proof that the system had generated. If there are errors in the student’s proof the system will give the student detailed feedback as to why their feedback is incorrect in a clear, easy-to-read and easy-to-learn format to aid in the learning of natural deduction.

Thusly, the project will be produced with the following steps:

- Visual design of the web-app

- Formula syntax validator

- Algorithm for generating the proof based on the given formula

- System for allowing the student to apply their rules to incomplete proofs

- System for comparing student and application proofs, and generating feedback

**Blog Link -** <https://liaisong.edublogs.org/>

This is where I will be writing updates about problems, solutions and results of anything related to the application.

**List of Achievable Objectives**

There are five main stages of the project that will be completed sequentially for the most part.

Visual design (simple):

* User will initially have one point of input: the formula
* A display of the full proof or a partial proof for the student to complete
* Feedback system of the student’s proof

Formula Syntax Validator (simple):

* Have a predefined set of characters allowed for input
* Letters and symbols for negation, conjunction, disjunction and implication ONLY

Proof-generating Algorithm (difficult):

* Retrieving and understanding the user’s formula
* Deduce each step in the proof using the predefined rules
* Structure and display the proof to the student in a format related to the course

Student Proof Entry (intermediate)

* Show an incomplete proof from the student’s input formula
* Allow the student to enter their own steps and apply the rules they wish

Application-Student Comparator (difficult)

* Compare the student’s proof to the application’s proof
* Identify where the student may have errors
* Return feedback in a clear and understandable format

**Survey of Related Work**

The main focus of the project is the ability to generate the proof from the given formula, thus I will be researching any systems that are able to generate these proofs. In addition, I will be conducting research on the natural deduction proof calculus with both resources gathered from my supervisor and reliable online sources.

There is an online platform that is capable of generating natural deduction proofs created by the Grenoble Informatics Laboratory (Liglab). Their system allows the user to enter their formula and generates a line-by-line proof, and can annotate each line detailing which rule was used along with the dependencies. I will be studying this system in order to aid both my understanding of natural deduction as part of an application and the creation of the proof generator. In addition the Liglab system has detailed pages of information on the natural deduction topic, including details of the different rules and how the system handles the formula syntax.

**Methodology**

Despite splitting the project into five distinct stages, nothing more than a traditional waterfall methodology is required, where after analysis of the project has been carried out, the software is designed, implemented and tested repeatedly. Afterwards evaluation of the application will take place, and then maintenance of the application. Each of these steps can be revisited from any point in the development stages.

Analysis of each stage will be carried out, where I will analyse what the application requires in order to attain the end goal and gather requirements for the completion of the current stage I am working on. This step in the methodology will include studying proof calculus and understanding how proofs can be translated to a software platform.

The next step is design. This is where I will be designing the layout of the visual display for the users and the software’s architecture.

The implementation of the software is where code-writing will occur.

After the implementation of the app based on the design, testing will occur. If there are any errors the design stage will be revisited to accommodate any bug fixes.

Evaluation of the software will occur after testing is successful. This involves receiving feedback from users and any changes will occur during maintenance of the software.

The Waterfall methodology is best for this project because it is disciplined by design, which is vital for the level of complexity of the natural deduction proof calculus. In addition, this methodology makes a point of ensuring the application is well documented at each phase of development, ensuring a high quality and high quantity of documentation throughout the project.

**Project Evaluation**

The project will be evaluated based on the usability of the system of the first-year students: the end-users.

End-users will be asked to enter their formulae into the application and observed to determine the effectiveness of the application as a teaching aid. Thus, students that were struggling with natural deduction in particular can be targeted as a group for evaluation of the software. The students will write their opinions of the application – such as ease-of-use, bug reporting, whether or not the system had helped them, and any changes they would make—which will be read over to help determine whether or not the application is effective.

**Project Plan**

Semester 1

Week 6 – Week 8 (23 Oct – 6 Nov)

* Analysis of the Liglab system and any other existing similar systems
* Requirements gathering for detailed specification of the project
* Discussing with second years about what they would have wanted for an app
* Researching proof calculus to develop a deeper understanding of ND
* Investigating software development tools and languages that suitable

Week 9 – Week 11 (13 Nov – 27 Nov)

* Design visual layout of the application
* Design software architecture based on investigation of software technologies
* Implementation of visual design
* Implementation of logic formula parser (syntax validator)
* Testing and documentation of logic formula parser

Christmas Vacation (4 Dec – 1 Jan)

* Implementation of proof generator
* Testing and documentation of proof generator

Semester 2

Week 1 – Week 4 (8 Jan – 5 Feb)

* Implementation of student-system proof checker
* Testing and documentation of student-system proof checker
* Create project poster (Fri 12 Jan deadline)

Week 5 – Week 7 (12 Feb – 26 Feb)

* Finish implementation
* Testing of system as a whole
* Deploy application for first year students to use
* Gather evaluation data from students

Week 8 – Week 10 (5 Mar – 19 Mar)

* Begin writing report

Week 11 (26 Mar)

* Submit report and source code

**Marking Scheme**

**Software Development-based Marking Scheme**

The software development marking scheme was chosen because the focus of the project is on the development of a web-based application to be used by first-year students. Thus the project will focus on the product and the resulting software.

|  |  |
| --- | --- |
| **Criteria** | **Marking Scheme** |
| Student Performance | 10 |
| Project Product | 35 |
| Project Process | 30 |
| Project Results and Evaluation | 15 |
| Report Presentation | 10 |

**Supervisor Comments (David Bevan)**

I have met with Lee and discussed the project. I am convinced that he understands the nature and scope of the project, and appreciates where the challenges lie.