Degree Audit Form

for Computer Engineering Students

Pranav Bheda - Julian Callin - Carlos Rivera

Abstract and Introduction

- SCU COEN students currently use eCampus degree audit
- Problems with the current solution
    - Single font
    - ASCII art formatting
    - One long page of text
    - No tables or interaction
- Our Single Page solution
    - Enter classes, see what has been fulfilled
    - Instant results
    - Easily readable
    - Persistence on machine

Functional Requirements

- Take a student’s input information regarding completion of degree requirements
- Tell students whether a major requirement is complete
- Tell students whether a core requirement is complete
- Tell students whether they possess extra class credits which can be used toward education enrichment requirements
- Persist between sessions
- Allow a student to print the degree report

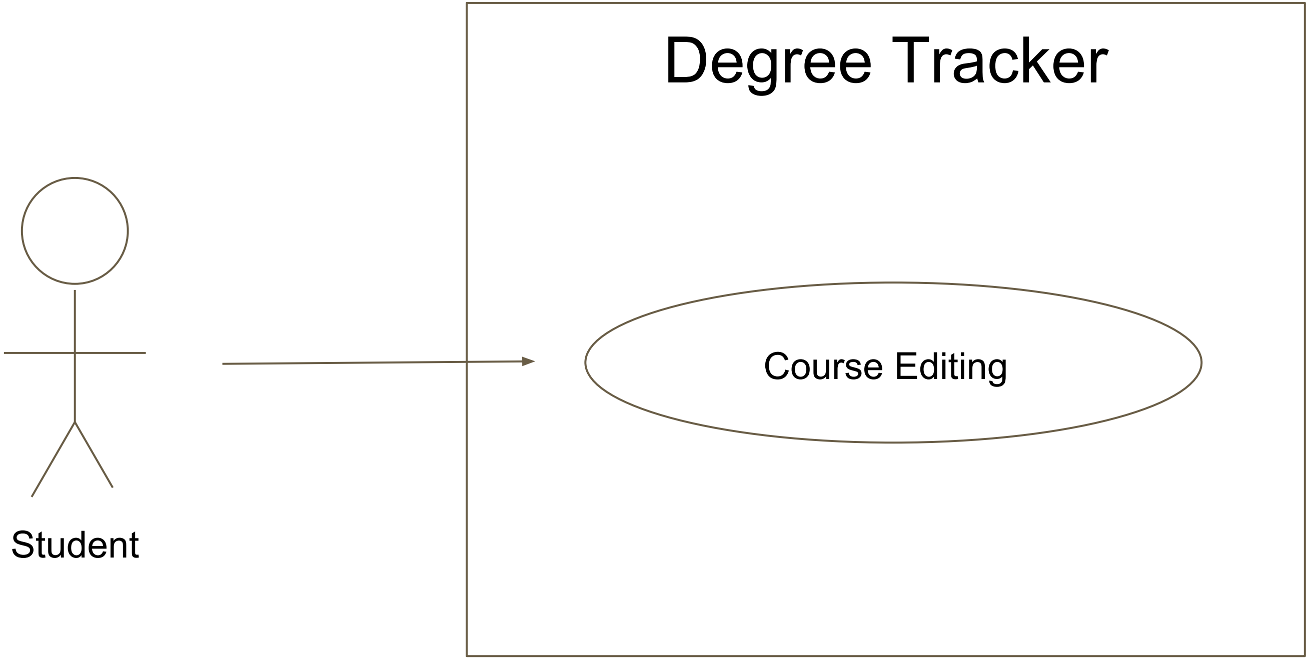
Non Functional Requirements

- User friendly
- Easily testable
- Easily maintainable

Design Constraints

- Must be web based
- Must run on the SCU ECC Linux and Windows machines
- Must retain functionality on Firefox and Chrome

Use Case Diagram



Activity Diagram

Technologies Used

HTML: A simple, universal web markup language

Bootstrap CSS: A framework for web design and formatting

JavaScript: A popular web language

Git: A method of source code control

Github: A host for source code and a collaborative environment

Architectural Diagram

Demo

link

Design Rationale

- One page UI
  - Eliminates navigation and scrolling
- Basic, lightweight web technologies
  - Good looking
  - Easy to run in any browser
- No relational database
  - Don’t need advanced queries
- Local Storage vs Cookies
  - Easier to implement local storage

Testing Procedure

- Our Local Development Computers
  - Unit testing with each course and what requirement it satisfies
  - Persistency
    - Closing windows, Opening new tabs
- Testing on ECC Linux and Windows machines
- Browser Testing (Firefox and Chrome)
- Testing using our own schedules
  - Verifying our system’s computed results with the eCampus Degree audit
- Testing using other students’ list of taken courses

Risk Analysis

Risk	Consequence	Probability	Severity	Impact	Mitigation Strategy
Run out of time	Cut Features	0.4	9	3.6	Ensure team members stay on schedule
Unexpected difficulty of feature implementation	Increased time to production	0.6	6	3.6	Allocate more time than necessary for component completion
Test cases will not completely cover all possible requirement combinations	User may experience unexpected behavior	0.3	8	2.4	Write unit tests or smoke tests using data, do more integration and user acceptance testing

Development Timeline

- Key Dates
  - Week 7 Demo
  - Week 9 Final Presentation
  - Week 10 Final Demo
- Design Weeks 4-6
  - Frontend - Pranav
  - Algorithm - Julian
  - Backend - Carlos
- Implementation Weeks 5-9
- Testing Weeks 5-10
  - Browser Capability
  - Algorithm Correctness
  - Ease of Use

Conclusion - Lessons Learned

- Project planning
- Work with frameworks you know well
- Start with small, then grow big
  - Test each course/requirement at a time, not all at the end
- Remember to “Keep it Simple Stupid”
  - Don’t include requirements that aren’t specified by customer

Thank you

Questions?