Studentable

*A University Course Registration Management System*

Early Design Document

*ETPB Software Design™ | 212 McAlister Dr, Saint John NB*

Table of Contents

[Preface 3](#_Toc57927498)

[Introduction 3](#_Toc57927499)

[Software Design method 4](#_Toc57927500)

# Preface

Within this document is the outline for an early version of a university course registration management system. This document will focus on the design aspects of this software system, primarily the technologies used and the decisions behind using those technologies. As this system evolves, so may this document. The intentions of *ETPB Software Design™* with this projectare to provide an example of how a software system should be built and what steps should be followed. It is purely educational and will in no way be used in a professional setting. All source code for the project will be located in the Github repository that also hosts this document.

# Introduction

Studentable is a web-based course registration management system. Its goal is to provide a user-friendly platform for students, professors and institutions to keep track of and organize their many courses and the students within them. The decisions made for the design of this system, which will be further explained later in the document, were all made to support an agile and simple product design. There is no need to overengineer something that students and institutions will rely on, therefore we opted for a method of development which was agile and allowed for continual improvement, but also validated our system at each step to make sure we were building the right system.

# Software Design method

The development of Studentable follows closely to the Extreme Programming (XP) approach. This is a pure agile method which focuses around a few fundamental best practices. For our purposes, we followed the following best practices when designing and developing out system:

* Small Releases:
  + We intent to release Studentable at first as a small system with its *essential* features (student/professor authentication, dashboard, database integration). Once this is accomplished, further versions will be released in short cycles which addon to the original release.
* Testing:
  + Testing will be a key factor when building this system. Due to the plethora of use cases a management system like this one brings forward; our team must ensure that we are always getting acceptable behavior from our software.
* Refactoring:
  + At first the system may be messy, but refactoring will be a routine task. This doesn’t necessarily add more to the software, but it does reduce complexity and improve quality which is of upmost importance.
* Continuous Integration:
  + The 3-Tier Architecture which was chosen for this project greatly supports the practice of continuous integration. When being developed, the system will be integrated and built many times a day to ensure we always have a working build.

Overall, the XP method may not be everyone’s first choice for a system like this. But with the unknown requirements that could possibly be brought forward by institutions who opt-into the system and the ease of communication with these institutions, the XP method will provide an approach that will allow us to quickly provide custom course management solutions to any University that hires us.