A Gamified Course Visualization, Organization, and Assessment System

Mark Floryan, a Computer Science professor at the University of Virginia, is dissatisfied with the way classes are currently taught. Floryan believes that strict deadlines and one-time assessments are not optimal for learning. Floryan seeks to understand the efficacy of gamification as an alternative method for learning. In his proposed system, students advance through a course as they advance through a video game: at their own pace, with deliberate practice and immediate feedback. Course material will be organized by topic, allowing for class progress to be visualized as a graph. Professors and students alike can see their progress in real time with on-site quizzes and automated grading.

Many classrooms today use the lecture-based exam model. In this model, lectures are the foundation of the course; quizzes, homework, and exams are all contingent on the lecture material. This solution forces students to learn at a controlled pace outside of the classroom, often causing students to move on from a topic before they have mastered it. Floryan's system aims to allow students to work at their own pace outside of the classroom and to make the mastery of a topic feel like beating a difficult boss in a video game.

Requirements are important because they enable clients to express their desires in a clear and unambiguous way. By converting client wishes to requirements and reviewing those requirements with customers, software developers can confirm that the product they will create matches the client's needs.

In the previous academic year, this project was started by a different team under the original set of requirements. Our contribution to the system includes adding security and

authentication, improving performance at scale, aligning the system with Floryan's course, and adding quizzes and auto-grading. Based on several meetings with Floryan, we assembled this list of requirements for our contribution:

Minimum Requirements:

- The professor must be able to access all functionality of the system
- The professor must be able to arrange topics to follow their syllabus
- The system must prevent student data from being accessed by anyone other than that student or course staff

Desired Requirements:

- The system must be able to display and automatically grade multiple choice questions
- The system must be able to display and automatically grade "Parson's Problem" questions
- The system must be able to generate quizzes from a professor-specified bank of questions
- The system must display course grades based on a topic-competency metric
- The system must be immediately responsive to students submitting assignments and viewing grades
- The system must be able to handle the uploading of large data sets by the professor in a reasonable time period
- The Professor must be able to give Teaching Assistants access to student grades and assignments
- Students should be able to view their own data and grades.
- Students should be able to view only topics which are currently unlocked.

Optional Requirements

- The professor must be able to set automatic grading procedures to match the grading system for a syllabus
- The system must be able to serve and grade (run) short-answer coding questions
- Graders must be able to manually assign grades to long-answer questions.

Figure 1.

Minimum Requirements:

- The professor must be able to access all functionality of the system
- The professor must be able to arrange topics to follow their syllabus
- The system must prevent student data from being accessed by anyone other than that student or course staff

Desired Requirements:

- The system must be able to display and automatically grade multiple choice questions
- The system must be able to display and automatically grade "Parson's Problem" questions
- The system must be able to generate quizzes from a professor-specified bank of questions

- The system must display course grades based on a topic-competency metric
- The system must be immediately responsive to students submitting assignments and viewing grades
- The system must be able to handle the uploading of large data sets by the professor in a reasonable time period
- The Professor must be able to give Teaching Assistants access to student grades and assignments
- Students should be able to view their own data and grades.
- Students should be able to view only topics which are currently unlocked.

Optional Requirements

- The professor must be able to set automatic grading procedures to match the grading system for a syllabus
- The system must be able to serve and grade (run) short-answer coding questions
- Graders must be able to manually assign grades to long-answer questions.

Figure 1.