

Tenjin Math

Amber Womack, Erik Eakins, Mike Watson

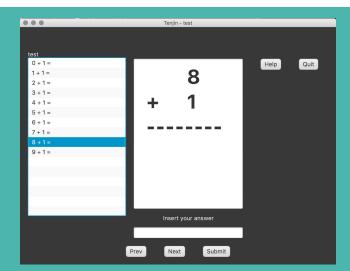
Our Project



Interactive program focusing on learning basic math skills with mathematical flashcards and workbooks.

DEMO TIME!!

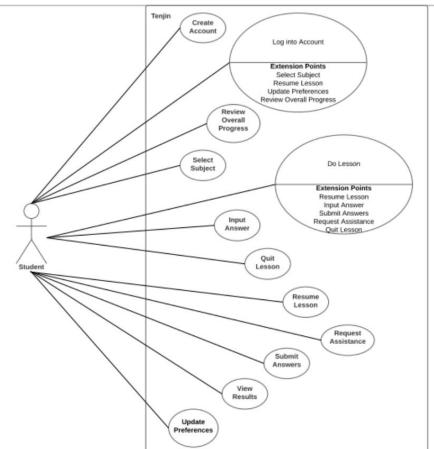
Demo Link: http://screencast.com/t/lUKbzoxb



TENJIN - USE CASE OVERVIEW

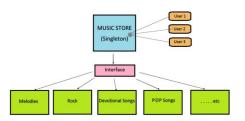
Use Cases - High Level

- Create Account/Login
- Select Module
- Start/Complete Lesson
- Quitting Lesson
- Logout



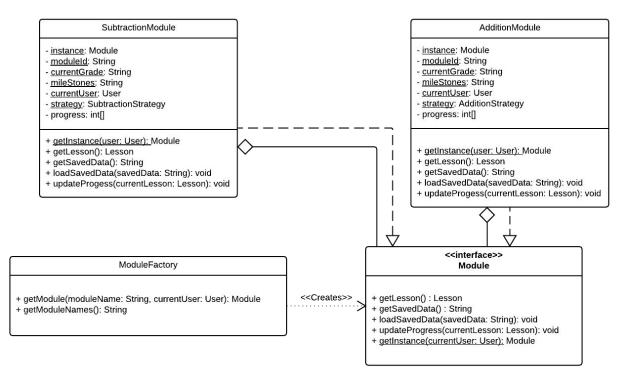
Design Patterns

- Factory
 - "Used to replace class constructors, by creating an abstraction through which one of several classes is returned determined at run-time."
 - Why: Used for loading a module based on the user selection
- Strategy
 - "Enables an algorithm's behavior to be selected at runtime."
 - Why: Used for building the lesson based off the user's grade progress
- Singleton
 - "Restricts the instantiation of a class to one object." useful when needing only one object to coordinate actions for whole system
 - Why: Used to make sure only 1 module was instantiated once for users



Class Diagram I

Factory Design Pattern



AdditionStrategyGradeK + generateLesson(): Lesson **Class Diagram II** AdditionStrategyGrade1 Strategy Design Pattern + generateLesson(): Lesson AdditionStrategyGrade2 + generateLesson(): Lesson AdditionStrategyGrade3 + generateLesson(): Lesson AdditionStrategyGrade4 + generateLesson(): Lesson AdditionStrategyGrade5 + generateLesson(): Lesson <<interface>> AdditionStrategy + generateLesson(): Lesson AdditionModule - instance: Module moduleld: String currentGrade: String mileStones: String currentUser: User strategy: AdditionStrategy ~ progress: int∏ + getInstance(user: User): Module + getLesson(): Lesson + getSavedData(): String + loadSavedData(savedData: String): void + updateProgess(currentLesson: Lesson): void

BE interested....

- PRE-design, Diagrams, Design Patterns Created ease of coding
- Class Diagram: most helpful
 - Helped with coding it out since it was more like fitting puzzle pieces together rather than making the pieces then trying to see if they will fit together
 - Didn't have class diagram for JavaFX
 - created a more difficult task to complete front end
 - More time consumption
 - More trial-and-error coding practices
- Systems used JavaFX

References & Credits

- https://play.google.com/store/apps/details?id=appinventor.ai_coolbhavaa1.
 mathcard_1
- http://danruizmath115.weebly.com/workbook.html
- https://en.wikipedia.org/wiki/Singleton_pattern
- http://www.w3programmers.com/category/php-design-patterns/
- Thank you to Professor Boese and teammates.