Joseph Nam
Derek Lu

<div align="center">Project 6</div>

State Summary
- School Scraping by Joseph Nam and Derek Lu
- Work Done
  - Scraped data from class search
  - Model from mvc mostly implemented
  - Skeleton code for other structures in class diagram done
  - Ideas on how to do scraping for rate my professor
  - Basic idea of how to implement GUI with Streamlit.
- Changes or Issues Encountered
  - Navigating html for tags to scrape
  - Taking scraped data and putting it into usable data structures
  - Separating responsibilities between model, view, and controller
  - Change - No longer using the observer pattern
- Patterns
  - MVC
    - The scraper and all the data gained from scraping will be the model that will be called from a controller which will also use a view object to display our GUI through Streamlit.
  - Iterator
    - We will use an Iterator pattern to iterate through all the Course objects we create from scraping from the CU class search. This will be a helpful tool in navigating through the course data.
  - Singleton
    - The Scraper_Model object will be used with the Singleton design pattern since we only need one scraper to interact with throughout our code.
  - Decorator
    - The Decorator pattern will be used within the GUI of the project which will change the color of the courses in the GUI to show red, green, or yellow based on professor score for that course.

Plan For Next Iteration
- Implement a streamlit GUI
  - Implement the View class
  - View will change dynamically based off of use input to the GUI, so the GUI needs to be able to interact with the controller
- Implement scraping for RateMyProfessors
  - Another functionality in the scraping_model class, called by the controller class
  - This will probably change/add data structures we end up using in the scraping_model class

Class Diagram

Model

ProfessorScore
- + overallScore: int
- + wouldTakeAgain: int
- + difficulty: int
- + ratingDistribution: int[ ]

ShoppingCart
- + courseList: Course [ ]
- + numCourses: int
- + displaySchedule( ): void
- + displayCart( ): void

Course
- + name: String
- + code: String
- + professor: String
- + subject: String
- + description: String
- + schedule: Datetime[ ]
- + inCart: bool
- + displayCourse( ): void
- + showSchedule( ): void

Professor
- + name: String
- + score: ProfessorScore
- + displayScores( ): void

Singleton Pattern

Scraper_Model
- + courses: Course [ ]
- + professors: Professor [ ]
- + instance: Scraper
- + iter: CourseIterator
- + cart: ShoppingCart
- + query( ): void
- - scrape_courses(String, String): void
- - scrape_profs(String): void
- - getInstance( ): Scraper

Iterator Pattern

CourseIterator
- + getNext( ): Course

Controller

Controller
- + model: Scraper_Model
- + model_set_PW(String): void
- + model_set_UN(String): void
- - scrape_courses(String, String): void
- - scrape_profs(String): void

Driver
- + scrape_model: Scraper_Model
- + controller: Controller
- + view: View
- + setup_model( ): void
- + run_scraping_instance( ): void

View

View
- + model: Scraper_Model
- + controller: Controller
- + courseDecor: ColorDecorator
- + display_gui( ): void
- + displayResults( ): void

Decorator Pattern

ColorDecorator
- + displayCourse( ): void
- + showSchedule( ): void

Green
- + displayCourse( ): void
- + showSchedule( ): void

Yellow
- + displayCourse( ): void
- + showSchedule( ): void

Red
- + displayCourse( ): void
- + showSchedule( ): void