CS62-Final-Project: TeamSync

Project Overview

> This program was built to help student-athletes manage their schedules by combining academic and athletic commitments into one system. We created a program that allows users to build schedules with academic and athletic events, detect conflicts, and filter courses based on department and number. The program includes a role-based access system so coaches can view their athletes' schedules. The program supports both student-athletes and coaches. It allows coaches to view their and their athlete's schedules, add events to the team and to individuals, add athletes, and more. Student-athletes can register for courses, view their profile and schedule, view conflicts, and add events. Users are connected to their practice schedule based on the athletic schedule that the coach uploaded. Using data from Fall 2024 courses at the Claremont College, courses are filtered by athletic schedule and users are able to filter by department and course number.

External Libraries

We did rely on external libraries for the development of this program.

Public API

Below is a summary of some of the public methods and constructors for the major classes in TeamSync. These methods represent the user-facing functionality of the application.

`Athlete`

Method Name and Inputs/Outputs	Description	Example Use
`Athlete(String name, String username, String team, String major, Schedule schedule, int gradYear)`	Constructs an `Athlete` object.	`new Athlete("Cecil", "cecil47", "Swim", "CS", new Schedule(), 2027)`
`void addEvent(Event e)`	Adds an event to the athlete's schedule.	`athlete.addEvent(event)`
`boolean hasConflicts()`	Returns true if the athlete has any overlapping events.	`athlete.hasConflicts()`
`ArrayList<Course> getEnrolledCourses()`	Returns the list of currently enrolled courses.	`athlete.getEnrolledCourses()`
`void enrollCourse(Course course)`	Adds a course and all its events to the athlete's schedule.	`athlete.enrollCourse(course)`

Coach

Method Signature	Description	Example Usage
`Coach(String name, String username, String team, Schedule schedule, HashMap<String, Athlete> athletes)`	Constructs a `Coach` object.	`new Coach("Coach Sagehen", "sagehen47" "Swim", schedule, athleteMap)`
`void addEventToTeam(Event sampleEvent)`	Adds an event to the coach and all athletes.	`coach.addEventToTeam(event)`
`void addAthlete(Athlete sampleAthlete)`	Adds an athlete to	`coach.addAthlete(athlete)`

Method Signature	Description	Example Usage
	the coach's team.	
`ArrayList<ArrayList<Event>> getAthleteConflicts(String username)`	Returns a list of event conflicts for a specific athlete.	`coach.getAthleteConflicts("kai456")`
`ArrayList<Event> createPracticeSchedule(String filePath, LocalDate start, LocalDate end)`	Reads a CSV and generates a semester-long practice schedule.	`coach.createPracticeSchedule("Data/sw start, end)`

`Schedule`

Method Signature	Description	Example Usage
`void addEvent(Event event)`	Adds an event and sorts the schedule.	`schedule.addEvent(event)`
`Event removeEvent(Event event)`	Removes the given event if found.	`schedule.removeEvent(event)`
`ArrayList<ArrayList<Event>> getConflicts()`	Returns all conflicting event pairs.	`schedule.getConflicts()`
`boolean detectConflict()`	Returns true if any overlapping events exist.	`schedule.detectConflict()`

`Event`

Method Signature	Description	Example Usage
`Event(dateTimePair start, dateTimePair end, String name, int type, String info)`	Constructs an event.	`new Event(start, end, "Practice", 2, "Pool")`

Method Signature	Description	Example Usage
`boolean detectOverlap(Event other)`	Checks if two events overlap in time.	`event1.detectOverlap(event2)`
`int getType()`	Returns the event type (1 = academic, 2 = athletic, 3 = other).	`event.getType()`
`String getName()`	Returns the name of the event.	`event.getName()`

`Course`

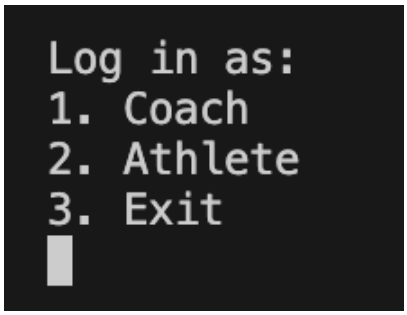
Method Signature	Description	Example Usage
`ArrayList<Event> courseToEvent()`	Converts the course into recurring class events.	`course.courseToEvent()`
`static ArrayList<Course> filterByDept(String prefix)`	Filters all courses by department prefix.	`Course.filterByDept("CSCI")`
`static ArrayList<Course> filterBySchedule(Schedule schedule)`	Returns only courses that don't conflict with a schedule.	`Course.filterBySchedule(schedule)`
`static ArrayList<Course> filterByBoth(String prefix, Schedule schedule)`	Filters courses by dept and conflict-freeness.	`Course.filterByBoth("CSCI", schedule)`

`dateTimePair`

Method Signature	Description	Example Usage
`dateTimePair(LocalDate date, LocalTime time)`	Constructs a new date-time pair.	`new dateTimePair(LocalDate.now(), LocalTime.now())`
`int compareTo(dateTimePair other)`	Compares two dateTimePairs.	`pair1.compareTo(pair2)`
`boolean equals(Object other)`	Checks equality of two dateTimePairs.	`pair1.equals(pair2)`
`String toString()`	String format `<date, time>`.	`System.out.println(pair)`

Running the code

Launching the Program



Run `MainProgram` to launch the TeamSync application. You'll be prompted to log in as a Coach or Athlete.

Logging in as the Coach

```
*** Coach Menu ***
1. View your schedule
2. Add event to team
3. View an athlete's schedule
4. View athlete conflicts
5. Input or change athletic schedule
6. View team roster
7. Add an athlete to the team
8. Go back
```

The coach can manage schedules, view conflicts, add athletes, and input an athletic practice schedule.

Uploading a Practice Schedule

```
*** Coach Menu ***
1. View your schedule
2. Add event to team
3. View an athlete's schedule
4. View athlete conflicts
5. Input or change athletic schedule
6. View team roster
7. Add an athlete to the team
8. Go back
5
Input athletic schedule filename.csv (Part of path after 'Data/')
SoccerSampleSchedule.csv
Input season start date in form YYYY-MM-DD
2024-08-15
Input season end date in form YYYY-MM-DD
2024-12-15
Athletic schedule uploaded.

*** Coach Menu ***
1. View your schedule
2. Add event to team
3. View an athlete's schedule
4. View athlete conflicts
5. Input or change athletic schedule
6. View team roster
7. Add an athlete to the team
8. Go back
```

After inputting the sample practice schedule file from the `Data/` folder and specifying the dates, the practice events are added to every athlete's schedule.

Switching to Athlete Login

```
*** Coach Menu ***
1. View your schedule
2. Add event to team
3. View an athlete's schedule
4. View athlete conflicts
5. Input or change athletic schedule
6. View team roster
7. Add an athlete to the team
8. Go back
8

Log in as:
1. Coach
2. Athlete
3. Exit
2
Enter your username: Guy
User not found. Please enter a username to create an account.
guy123
Enter your name.
Guy Fuchs
Enter your team
Soccer
Enter your major
CS
Enter your grad year
2027
Account created.

*** Athlete Menu (Guy Fuchs) ***
1. View full profile and semester schedule.
2. View conflicts
3. Add event
4. Register for courses
5. Back
```

If the username is not found, you'll be prompted to create an account. This registers the user and adds them to the coach's list of athletes. This shows the role-based access feature.

Course Registration with Filtering

```
*** Athlete Menu (Guy Fuchs) ***
1. View full profile and semester schedule.
2. View conflicts
3. Add event
4. Register for courses
5. Back
4
You currently have no academic courses.

What would you like to do?
1. Register for academic courses
2. Go back to the previous menu
1
Course filter options
1. Filter by department
2. Filter by athletic schedule
3. Filter by both
3
Input department prefix
CSCI
How many courses do you want to want to register for? Pick between 1 and 5.
2
```

Athletes can register for courses by filtering by academic department, the coach-inputted athletic schedule, or both. In this case, the athlete filters by both. This allows for the viewing of classes based on input parameters.

Adding Courses to Schedule

```
How many courses do you want to want to register for? Pick between 1 and 5.
2
1: Course ID: CSCI005  HM-01 FA2024, Time: 935-1050, Days: --T-R--, Location: HM GA MCAL
2: Course ID: CSCI005  HM-02 FA2024, Time: 935-1050, Days: --T-R--, Location: HM SHAN 2475
3: Course ID: CSCI005L HM-01 FA2024, Time: 1445-1645, Days: --T----, Location: HM MCSC 203
4: Course ID: CSCI005L HM-01 FA2024, Time: 1445-1645, Days: --T----, Location: HM MCSC 204
5: Course ID: CSCI005L HM-01 FA2024, Time: 1445-1645, Days: --T----, Location: HM MCSC 205
6: Course ID: CSCI005L HM-02 FA2024, Time: 1445-1645, Days: ----R--, Location: HM MCSC 203
7: Course ID: CSCI005L HM-02 FA2024, Time: 1445-1645, Days: ----R--, Location: HM MCSC 204
8: Course ID: CSCI005L HM-02 FA2024, Time: 1445-1645, Days: ----R--, Location: HM MCSC 205
9: Course ID: CSCI036  CM-01 FA2024, Time: 1615-1730, Days: -M-W---, Location: CM KRV 165
10: Course ID: CSCI036  CM-02 FA2024, Time: 900-950, Days: -M-W-F-, Location: CM KRV 165
11: Course ID: CSCI036P PZ-01 FA2024, Time: 1315-1430, Days: --T-R--, Location: PZ AV 201
12: Course ID: CSCI040  CM-01 FA2024, Time: 1445-1600, Days: -M-W---, Location: CM RN 12
13: Course ID: CSCI040  CM-02 FA2024, Time: 1615-1730, Days: -M-W---, Location: CM RN 12
14: Course ID: CSCI042  HM-01 FA2024, Time: 935-1050, Days: --T-R--, Location: HM SHAN 2460
```



```
Input number next to course you want to add
27
Enrolled in:
 Course ID: CSCI062  PO-01 FA2024, Time: 1315-1430, Days: -M-W---, Location: PO SCOM 102
Input number next to course you want to add
46
Enrolled in:
 Course ID: CSCI131  HM-01 FA2024, Time: 935-1050, Days: --T-R--, Location: HM MCSC 203
Course registration complete.

*** Athlete Menu (Guy Fuchs) ***
1. View full profile and semester schedule.
2. View conflicts
3. Add event
4. Register for courses
5. Back
```

Select courses by entering the number shown next to each listing. Courses are added to the athlete's schedule instantly. The image of courses above shows only a small portion of the courses that will show up when using this feature.

Additionally, since courses were filtered using the coach-inputted schedule, the athlete currently has no conflicts.

Conflict Detection

```
Course registration complete.

*** Athlete Menu (Guy Fuchs) ***
1. View full profile and semester schedule.
2. View conflicts
3. Add event
4. Register for courses
5. Back
3
Event name: Movie
Date (YYYY-MM-DD): 2024-10-14
Start time (HH:MM): 16:30
End time (HH:MM): 19:30
Event type (1 = Academic, 2 = Athletic, 3 = Other): 3
Extra info (e.g., location): Edmunds Ballroom
Event added.

*** Athlete Menu (Guy Fuchs) ***
1. View full profile and semester schedule.
2. View conflicts
3. Add event
4. Register for courses
5. Back
2
Conflicts for Guy Fuchs:
Conflict 1:
        <2024-10-14, 16:30> - <2024-10-14, 19:30>, Movie (Other);
        <2024-10-14, 17:00> - <2024-10-14, 19:00>, Practice (Athletic);

*** Athlete Menu (Guy Fuchs) ***
1. View full profile and semester schedule.
2. View conflicts
3. Add event
4. Register for courses
5. Back
```

An athlete adds a new event that overlaps with a class. The program is able to detect this conflict, demonstrating our conflict-detection feature alongside our adding event feature, which both the athlete and coach can do.

Viewing Athlete Profile

```
*** Athlete Menu (Guy Fuchs) ***
1. View full profile and semester schedule.
2. View conflicts
3. Add event
4. Register for courses
5. Back
1


Athlete: Guy Fuchs
Team: Soccer
Major: CS
Grad Year: 2027
Schedule:
<2024-08-15, 16:30> - <2024-08-15, 17:30>, Practice (Athletic);
<2024-08-15, 17:45> - <2024-08-15, 18:45>, Film (Athletic);
<2024-08-16, 17:00> - <2024-08-16, 19:00>, Practice (Athletic);
<2024-08-17, 19:00> - <2024-08-17, 21:00>, Game (Athletic);
<2024-08-18, 12:00> - <2024-08-18, 13:00>, Practice (Athletic);
<2024-08-19, 17:00> - <2024-08-19, 19:00>, Practice (Athletic);
<2024-08-20, 17:00> - <2024-08-20, 19:00>, Practice (Athletic);
<2024-08-21, 16:30> - <2024-08-21, 18:30>, Practice (Athletic);
<2024-08-22, 16:30> - <2024-08-22, 17:30>, Practice (Athletic);
<2024-08-22, 17:45> - <2024-08-22, 18:45>, Film (Athletic);
<2024-08-23, 17:00> - <2024-08-23, 19:00>, Practice (Athletic);
<2024-08-24, 19:00> - <2024-08-24, 21:00>, Game (Athletic);
<2024-08-25, 12:00> - <2024-08-25, 13:00>, Practice (Athletic);
<2024-08-26, 13:15> - <2024-08-26, 14:30>, CSCI062  PO-01 FA2024 (Academic);
<2024-08-26, 17:00> - <2024-08-26, 19:00>, Practice (Athletic);
<2024-08-27, 09:35> - <2024-08-27, 10:50>, CSCI131  HM-01 FA2024 (Academic);
<2024-08-27, 17:00> - <2024-08-27, 19:00>, Practice (Athletic);
<2024-08-28, 13:15> - <2024-08-28, 14:30>, CSCI062  PO-01 FA2024 (Academic);
<2024-08-28, 16:30> - <2024-08-28, 18:30>, Practice (Athletic);
<2024-08-29, 09:35> - <2024-08-29, 10:50>, CSCI131  HM-01 FA2024 (Academic);
<2024-08-29, 16:30> - <2024-08-29, 17:30>, Practice (Athletic);
<2024-08-29, 17:45> - <2024-08-29, 18:45>, Film (Athletic);
```

The athlete can view their full profile and semester schedule. The image cuts off part of the athlete's schedule for the whole semester.

Coach View of Athlete Conflicts

```
*** Coach Menu ***
1. View your schedule
2. Add event to team
3. View an athlete's schedule
4. View athlete conflicts
5. Input or change athletic schedule
6. View team roster
7. Add an athlete to the team
8. Go back
4
Conflicts for athlete1:
None
Conflicts for Guy Fuchs:
Conflict 1:
        <2024-10-14, 16:30> - <2024-10-14, 19:30>, Movie (Other);
        <2024-10-14, 17:00> - <2024-10-14, 19:00>, Practice (Athletic);

*** Coach Menu ***
1. View your schedule
2. Add event to team
3. View an athlete's schedule
4. View athlete conflicts
5. Input or change athletic schedule
6. View team roster
7. Add an athlete to the team
8. Go back
6

The athletes on your team are:

Athlete: athlete1
Grad Year: 2027

Athlete: Guy Fuchs
Grad Year: 2027
```

The coach can view all team conflicts. In this example, only "Guy" has a conflict, while the auto-added "athlete1" does not.

Additional Features

- Coaches can view the athlete roster
- You can experiment with adding/removing events, filtering courses, or printing team data

Final Note and Future Improvements

In the future, we want to expand this program to allow for multiple teams or even universities. Additionally, there are many methods written in the source code that were not used when writing the main program. These additional methods would allow for a more extensive application and provide many additional features for the users. Some of these methods include, but are not limited to, a coach adding an event to a single athlete's schedule and removing an event from a schedule.

For now though, we hope you enjoy what we have built. Have fun navigating TeamSync!

Authors

Kai Parker, Guy Fuchs, and Tiernan Colby