

Sprint 3 Retrospective

Contents

Team Roles.....	1
Dates of Sprint.....	2
Information about Team and Contributions.....	2
Sprint Goal.....	3
Sprint Achievements.....	3
Sprint Backlog Items and Status.....	3
Burndown and Burnup Chart.....	4
Design Diagram.....	6
Documentation of changes.....	7
Evaluation of code and test quality.....	7
Customer Meeting.....	12
Links.....	13

Team Roles

Product Owner: Yuqi Fan

Scrum Master: Pavithra Gopalakrishnan

Dates of Sprint

4th November 2024 - 15th November 2024

Information about Team and Contributions

Team Member	Time	Points	Role	Contribution
Yuqi Fan	2.5 Days	5	Product Owner	Yuqi worked to resolve the hosting of the algorithm on Heroku as well as being able to properly generate a schedule by uploading data. He also worked directly with the client to set up meetings, set agendas, and ensure that client needs were met.
Pavithra Gopalakrishnan	2 Days	4	Scrum Master	Pavithra coordinated with the team and the product owner and ensured the developers were clear on the evolving requirements. She also took care of documenting the efforts and conducting sprint ceremonies. Additionally, she finished fleshing out the time slots blocking feature and helped with UI/bug fixes.
Navya Unnikrishnan	2 Days	4	Developer	Navya worked on handling preferences with similar names between courses and instructors and worked to ensure that users could only see their own schedules. She also worked on redesigning the UI and ensuring all the test cases passed.
Navya Priya Nandimandalam	2 Days	4	Developer	Navya worked on the locking course functionality as well as exporting schedules as a CSV after generation. She also followed the TDD and BDD approach to develop the feature.

Colby Endres	2.5 Days	5	Developer	Colby worked on incorporating instructor preferences into the algorithm, removing constraints that force professors and courses to be equal, and working on generating schedules by uploading data. He also followed Test-driven development.
Wahib Sabir Kapdi	2.5 Days	5	Developer	Wahib fixed a bug relating to adding room booking tabs and also worked with Navya on locking courses and time slots after schedule generation. He also allowed for the hiding of courses from the generator and incorporated section ID and Number Fixes in JS. He also made sure to cover all test cases and meet the acceptance criteria.
Abel Gizaw	2 Days	3	Developer	Abel worked on trying to integrate our various soft constraints and ensuring that only allowed instructors should be added to a room booking. Additionally, he assisted in writing steps for cucumber tests.

Team Member	Total Points
Yuqi Fan	$2+3+1+5.5 = 11.5$
Pavithra Gopalakrishnan	$4+2+2+5 = 13$
Navya Unnikrishnan	$4+3+3+3 = 13$
Navya Priya Nandimandalam	$4+2+4+3 = 13$
Colby Endres	$2+2+4+4.5 = 12.5$
Wahib Sabir Kapdi	$4+3+5+4 = 16$
Abel Gizaw	$2+2+1+4 = 9$
Total Team Points	88

Sprint Goal

The goal of this sprint was to enhance the schedule generation system by implementing several critical features, including integrating instructor preferences, improving the user interface, enabling schedule export functionality, and ensuring proper scheduling logic through various constraints and blocking mechanisms. Additionally, the team focused on refining user access control and improving bug tracking and resolution to deliver a stable, fully functional system ready for client use.

Sprint Achievements

- Successfully resolved hosting issues related to the algorithm on Heroku.
- Enabled the ability to generate schedules by uploading data.
- Developed the locking course functionality
- Incorporated instructor preferences into the algorithm
- Added functionality to hide courses from the generator
- Integrated various soft constraints into the system
- Enabled a user view where a user sees only their schedules

Sprint Backlog Items and Status

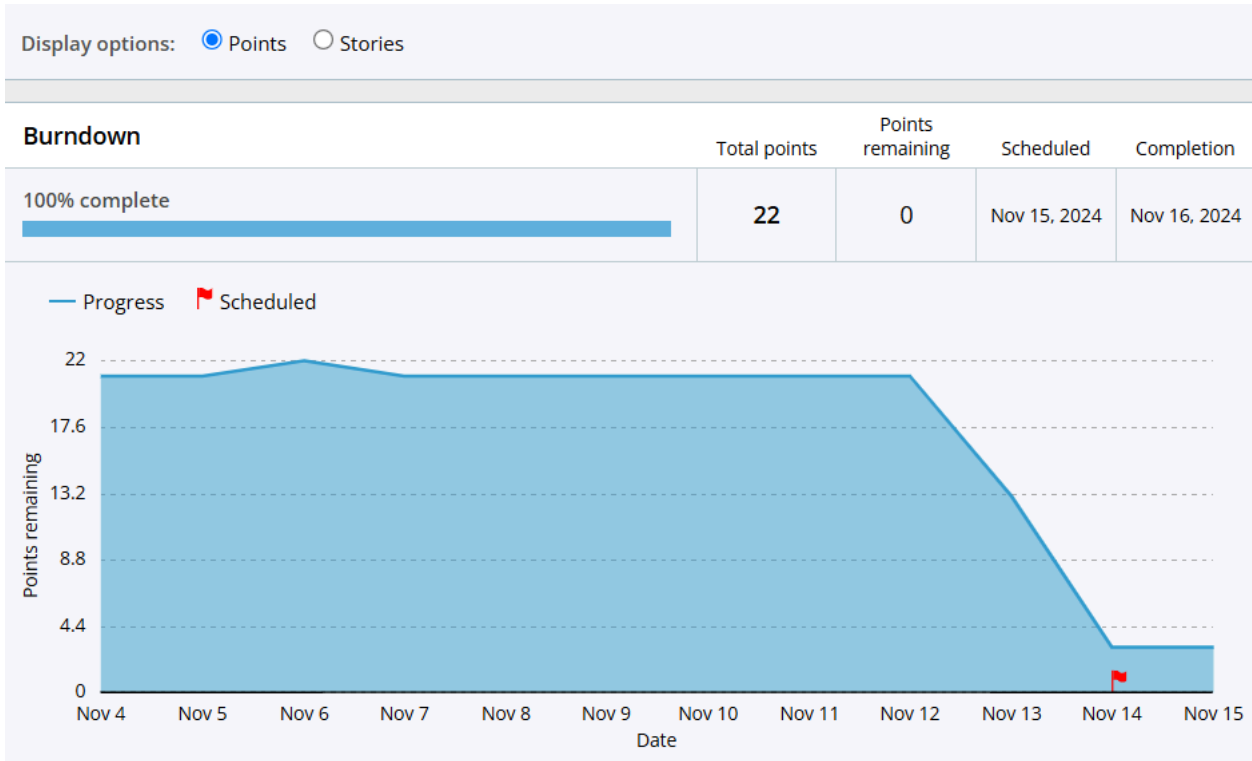
Change	Story/Bug	Status
+	Time Slot blocking	Completed
+	New Room Booking Bug	Completed
+	Courses / Time Slot locking	Completed
+	Generator Course Hiding Functionality	Completed
+	Similar Name Preferences Handling	Completed
+	Algorithm Heroku Hosting	Completed
+	Generate schedule by uploading csv	Completed

+	Export created schedules	Completed
+	User specific schedule views	Completed
+	Incorporate instructor preference into algorithm	Completed
+	Add allowed instructors to room booking	Completed
+*	Added lock and unlock icons	Completed
+*	Return from hidden courses bug fix	Completed
+*	Treating of timing as a hard constraint in the algo	Completed

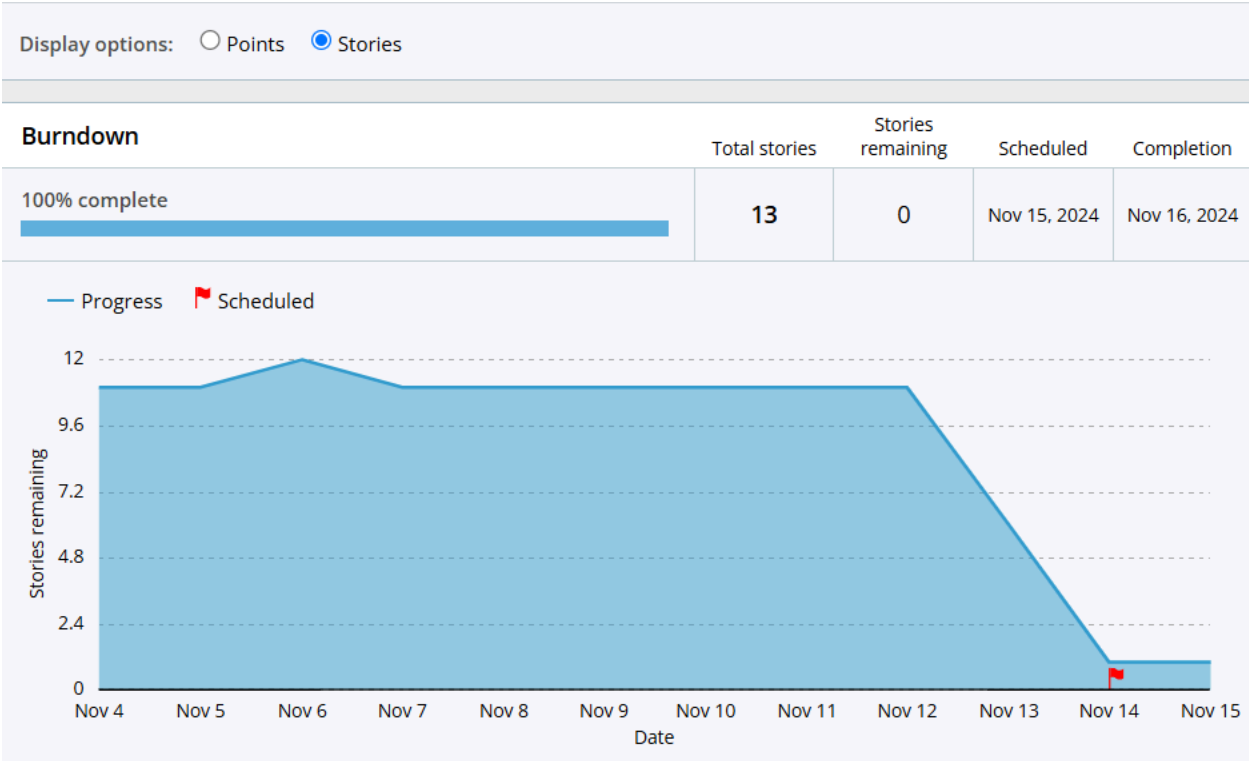
*Implemented after the end of the sprint as per the client’s request

Burndown and Burnup Chart

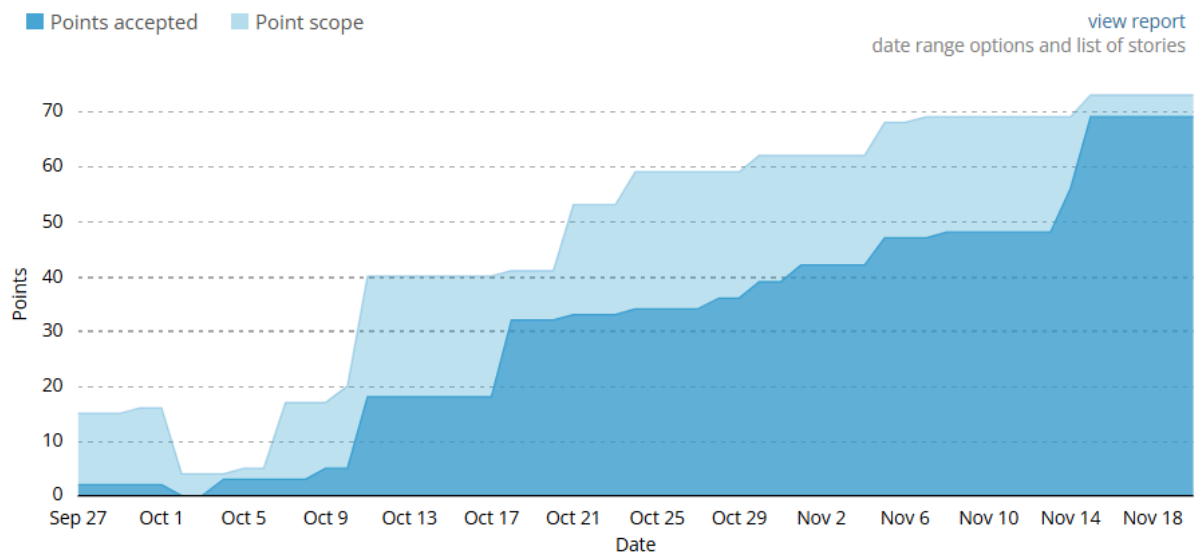
Point-based burndown chart for this sprint



Story-based burndown chart for this sprint



Sprint burnup chart



Design Diagram

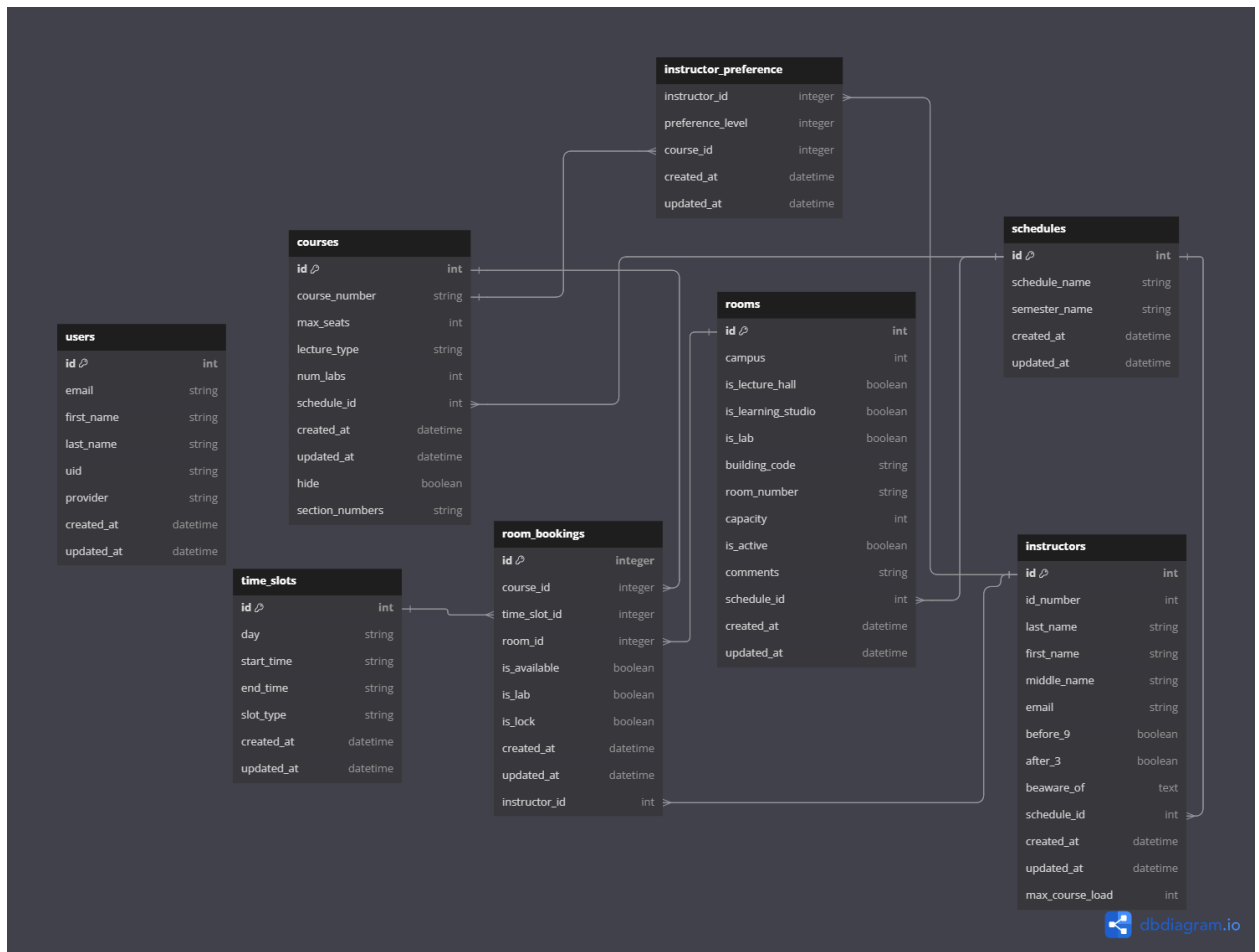
UI Design

- No changes were made

- <https://www.figma.com/design/CRRqUd8c0q8BnKWkPIY1pS/AggieAssign?node-id=1-2377&node-type=frame>

DB Design

- Sections table removed
- The max_course_load column was added to the instructor table
- The hide column was added to the course table
- Is_lock was added to the room_bookings table



Documentation of changes

DB Design

- Sections table removed
- The max_course_load column was added to the instructor table
- The hide column was added to the courses table
- Is_lock was added to the room_bookings table

Mock-ups and storyboards

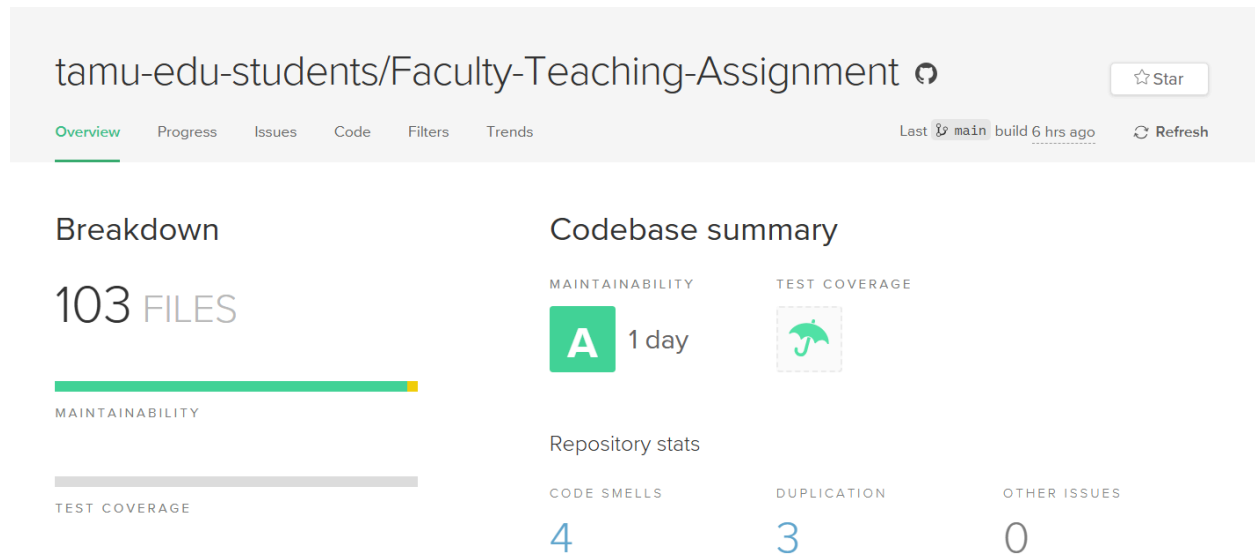
- There was no update.

Team working agreement Updates

- After merging any code to main, manually test the code and try to break it.

Evaluation of code and test quality

Code Climate



Rubocop

```

spec/controllers/schedules_controller_spec.rb:5:1: C: Metrics/BlockLength: Block has too many lines. [167/150]
RSpec.describe SchedulesController, type: :controller do ...
~~~~~
139 files inspected, 45 offenses detected

Tip: Based on detected gems, the following RuboCop extension libraries might be helpful:
* rubocop-capbara (https://rubygems.org/gems/rubocop-capbara)
* rubocop-factory_bot (https://rubygems.org/gems/rubocop-factory_bot)
* rubocop-rspec (https://rubygems.org/gems/rubocop-rspec)
* rubocop-rspec_rails (https://rubygems.org/gems/rubocop-rspec_rails)

The following RuboCop extension libraries are installed but not loaded in config:
* rubocop-rails

You can opt out of this message by adding the following to your config (see https://docs.rubocop.org/rubocop/extensions.html#extension-suggestions for more options):
AllCops:
  SuggestExtensions: false
pgopal1719@PavithraLenovo:~/CSCE606/Faculty-Teaching-Assignment$

```

Coverage

Unit Tests (rspec)

```

Finished in 3.91 seconds (files took 3.51 seconds to load)
119 examples, 0 failures

Coverage report generated for RSpec to /home/pgopal1719/CSCE606/Faculty-Teaching-Assignment/coverage.
Line Coverage: 94.79% (655 / 691)
pgopal1719@PavithraLenovo:~/CSCE606/Faculty-Teaching-Assignment$

```


Integration Tests (cucumber)

```
Share your Cucumber Report with your team at https://reports.cucumber.io


Command line option:  --publish
Environment variable:  CUCUMBER_PUBLISH_ENABLED=true
cucumber.yml:         default: --publish

More information at https://cucumber.io/docs/cucumber/environment-variables/

To disable this message, specify CUCUMBER_PUBLISH_QUIET=true or use the
--publish-quiet option. You can also add this to your cucumber.yml:
default: --publish-quiet

Coverage report generated for Cucumber Features, RSpec to /home/pgopal719/CSCE606/Faculty-Teaching-Assignment/coverage.
Line Coverage: 96.96% (670 / 691)
pgopal719@PavithraLenovo:~/CSCE606/Faculty-Teaching-Assignment$
```

Tests Covered as part of Sprint 3

 cucumberTests.mp4

Unit Test Cases Covered:

Instructors Controller

- incorrect schedule id: Checks the invalid case
- without added instructors: When no instructors are added to a schedule, display an appropriate message
- with added instructors: Happy case, when instructors are added to a schedule
- Fetches correct preferences for an instructor

Rooms Controller

- incorrect schedule id: Checks the invalid case
- without added rooms: When no rooms are added to a schedule
- with added rooms: Happy case, when rooms are added to a schedule
- without any filters or sorting: When no filters or sort order is chosen
- with active filter: When filters are added
- with sorting by building_code in ascending order: Test sort by building_code
- with sorting by capacity in descending order: Test sort by capacity

SchedulesController

- GET #index: populates an array of schedules
- GET #show: assigns the requested schedule and renders the template

- GET #new: assigns a new schedule
- POST #create: saves new schedule to DB
- DELETE #destroy: delete schedule from DB
- POST #upload_rooms: upload rooms for a particular schedule
- POST #upload_instructors: upload instructors for a particular schedule

TimeSlotsController

- assigns all time slots when no filters are applied: display all time slots
- filters time slots by day: filter by day
- filters time slots by type: filter by type
- filters time slots by day and type: filter by day and type

Room_booking_controller

- GET #index: verifies functionality for existing room booking
- POST #create: saves new schedule to DB
- DELETE #destroy: delete schedule from DB

Courses_controller

- GET #index
 - With valid schedule ID and no hidden courses, verifies that all visible courses are displayed for the schedule.
 - With valid schedule ID and hidden courses shown, ensures that hidden courses are displayed when show_hidden is true.
 - With invalid schedule ID, ensures the user is redirected with an appropriate alert.
 - With sorting by course_number in ascending order, verifies courses are sorted by course_number in ascending order.
- POST #toggle_hide
 - Hides a course with no associated room bookings, ensures the course is hidden successfully and the user is redirected with a success notice.
 - Fails to hide a course with associated room bookings, verifies the course remains visible, and shows an alert message.
 - Unhides a course successfully, and ensures a previously hidden course is made visible.
 - With an invalid course ID, it verifies that an appropriate error is displayed when the course is not found.

Customer Meeting

- 1) 1:45 PM November 13, 2024
Place: PETR 102B

Customer Meeting Details and Feedback

Presented and showcased the current MVP using actual data provided by the client. Allowed the client to mentally walk through his requirements alongside our implementation to clarify any confusion. Gathered small changes to implement in the coming sprint and clarified doubts pertaining to conflicting courses and instructors' CSV files.

2) 1:45 PM November 20, 2024

Place: PETR 102B

Customer Meeting Details and Feedback

Presented and showcased our final project and allowed the professor to see a start-to-end demo of the product. Answered all questions pertaining to actual usage and discussed areas for future improvement. Gathered general feedback on our team's work the entire semester.

Links

Deployed App: <https://faculty-teaching-assignment-31f5f9c405bc.herokuapp.com/>

GitHub Repository: <https://github.com/tamu-edu-students/Faculty-Teaching-Assignment>

Pivotal Tracker: <https://www.pivotaltracker.com/n/projects/2721604>

Slack: <https://tamu.slack.com/archives/C07PA043PA7>

Team Working agreement:

<https://github.com/tamu-edu-students/Faculty-Teaching-Assignment/blob/main/documentation/Fall2024/Team%20Working%20Agreement.md>

Code Climate Report:

<https://codeclimate.com/github/tamu-edu-students/Faculty-Teaching-Assignment>