BoilerBot

Team 1: Aditya Dhingra, Rahul Pai, Eehita Parameswaran,
Kush Rustagi, Devaunsh Sambhav, Yash Shiroya

May 1st, 2017

Sprint 3 Retrospective

Contents

1 What went well?

1.1 UserStory #19: As a user, I would like bot to provide me with information about the opening and closing times of dining courts at Purdue.

Status: Completed.

1.1.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save dining courts information to the database .
3. Given that the user requests opening and closing times of dining courts, we should be able to access the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get opening and closing times of dining courts at Purdue when the user requests for such information.

1.1.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested opening and closing times of dining courts, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received the opening and closing times of dining courts at Purdue.

1.2 UserStory #20: As a user, I would like the bot to provide me with information about food at dining courts at Purdue.

Status: Completed.

1.2.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save dining courts menu information to the database .
3. Given that the user requests food at dining courts, we should be able to access the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get food at dining courts Purdue when the user requests for such information.

1.2.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested dining courts menu, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received the the dining courts menu at Purdue.

1.3 UserStory #22: As a user, I want the bot to have information about the courses being offered in the current semester.

Status: Completed.

1.3.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save course information to the database .
3. Given that the user requests courses being offered in the current semester, we should be able to access the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get courses being offered in the current semester at Purdue when the user requests for such information.

1.3.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested courses offered in current semester, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received the courses offered in current semester.

1.4 UserStory #23: As a student at Purdue, I would like to know the seats left in a course that I want to register for.

Status: Completed.

1.4.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save seats left in a course information to the database .
3. Given that the user requests seats left in a course, we should be able to access the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get seats left in a course at Purdue when the user requests for such information.

1.4.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested seats left in a course, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received the seats left in a course.

1.5 UserStory #24: As a student at Purdue, I would like the bot to provide me with information about the different professors teaching a course.

Status: Completed.

1.5.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save information about different professors teaching a course to the database .
3. Given that the user requests information about different professors, we should be able to retrieve the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get information about different professors teaching a course at Purdue when the user requests for such information.

1.5.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested information about different professors, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received information about different professors.

1.6 UserStory #25: As a student at Purdue, I would like the bot to provide me with information about the different rooms that a specific course will be taught in.

Status: Completed.

1.6.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save information about different rooms of a specific course to the database .
3. Given that the user requests information about different rooms for a specific course, we should be able to retrieve the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get information about different rooms of a specific course at Purdue when the user requests for such information.

1.6.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested rooms for a particular course, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received the rooms for a particular course.

1.7 UserStory #26: As a student at Purdue, I would like the bot to provide me with information about the different times at which a specific course is being offered.

Status: Completed.

1.7.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save information about different times of a specific course to the database .
3. Given that the user requests information about different times for a specific course, we should be able to retrieve the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get information about different times of a specific course at Purdue when the user requests for such information.

1.7.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested different timings of a course, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received the different timings of a course.

1.8 UserStory #27: As a student at Purdue, I would like the bot to give me details like the credits offered for taking a course.

Status: Completed.

1.8.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, then it would be sent to Recast.ai for parsing.
2. Given that we can access data from the Purdue.io page, we would be able to save credits offered for taking a course to the database .
3. Given that the user requests information about credits offered for taking a course, we should be able to retrieve the correct details from the database.
4. Test feature by using Mocha-Chai testing framework and check whether or not we get credits offered for taking a course at Purdue when the user requests for such information.

1.8.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully extracted information from the Purdue.io page and then saved the details to the database.
3. When user requested credits offered for a particular course, then our queries accessed the correct information from our database.
4. We tested feature using automated testing frameworks and ensured that user received the credits offered for taking a particular course.

1.9 UserStory #28: As a user, I want to be able to submit features that I would like to see on the bot.

Status: Completed.

1.9.1 Acceptance Criteria

1. Given that our Natural Language Processing algorithm has been modified, when a user sends a message, then it would be sent to Recast.ai for parsing.
2. Given that Recast.ai is set up, run queries such that the feedback gets saved to the database schema.
3. Test feature by using Mocha-Chai testing framework and check whether or not our database is saving user suggested feedback.

1.9.2 Validation

1. We ensured that when the user sent a message, it was sent to Recast.ai, and then the intent we received from our Natural Language Processing algorithm was correct.
2. We successfully ran queries when recast.ai was set up and then made sure that the feedback was getting saved to the database schema.
3. We tested feature using automated testing frameworks and ensured that database saved user suggested feedback.

1.10 DeveloperStory #4: As a developer, I would like to be able to handle the UI for mobile and web appropriately.

Status: Completed.

1.10.1 Acceptance Criteria

1. Test feature manually using Messenger and by adding multiple users as testers to ensure that our bot handles UI for mobile, web appropriately.

1.10.2 Validation

1. We tested our features manually using the Facebook Developers platform and successfully ensure that mobile-web were handled appropriately.

1.11 DeveloperStory #5: As a developer, I would like to implement further fixes and enhancements according to the usage of the user.

Status: Completed.

1.11.1 Acceptance Criteria

1. Given that the users are providing feedback to an image, when the user rates an image, then store the rating in the database.
2. Given that the feedback is stored in the database, then analyze the feedback for further enhancements of filter recommendations to the user.
3. Test feature using stored procedure scripts to ensure that feedback model in the database is correct and can be analyzed.

1.11.2 Validation

1. We ensured that when user provides some rating, then it is stored in the database.
2. We successfully analyzed the feedback using an algorithm to provide enhancements to the user.
3. We tested the feature using stored procedure scripts and ensure that the feedback model in the database can be analyzed.

1.12 DeveloperStory #6: As a developer, I would like to be able to implement popular feature requests submitted by students.

Status: Completed.

1.12.1 Acceptance Criteria

1. Given that have feedback in our database model, then when user suggests feature, our algorithm should be able to parse it.
2. Given that we have an algorithm that can parse feedback, then the code should be able to identify the most frequent user request.
3. Test feature using stored procedure scripts to ensure that algorithm is identifying the most common feature request.

1.12.2 Validation

1. We successfully checked that our algorithm was able to parse feedback when user suggested feedback.
2. Our code was able to identify frequent user request when we had to parse the feedback.
3. We tested the feature using stored procedure scripts and ensured that algorithm was identifying most common feature request.

2 What did not go well?

We were able to meet all our targets and even implemented a couple of auxiliary features so if anything, we wish we had better time management. This is because quite a few times, our meetings went on for too long as we would brainstorm all possible ways of achieving a particular feature. We should have instead focussed on getting tasks done as soon as possible.

3 What will we do better in the next sprint?

We completed all our user stories/product backlog in its entirety and we are quite happy with our performance in this particular sprint! So, with Sprint 3 we completed all our tasks.