**Beaker Use Cases**

Jay Patel

Christian Klein

Joseph Thill

Derek Yu

Zifeng Jiang

**Use Case 1**

*Use Case Name:* New Previous Cell

*Actor:* User

*Goal:* The user wants to create a new cell of the language in the cell above the button.

*Preconditions:* A beaker notebook has been created or opened.

*Main Flow:*

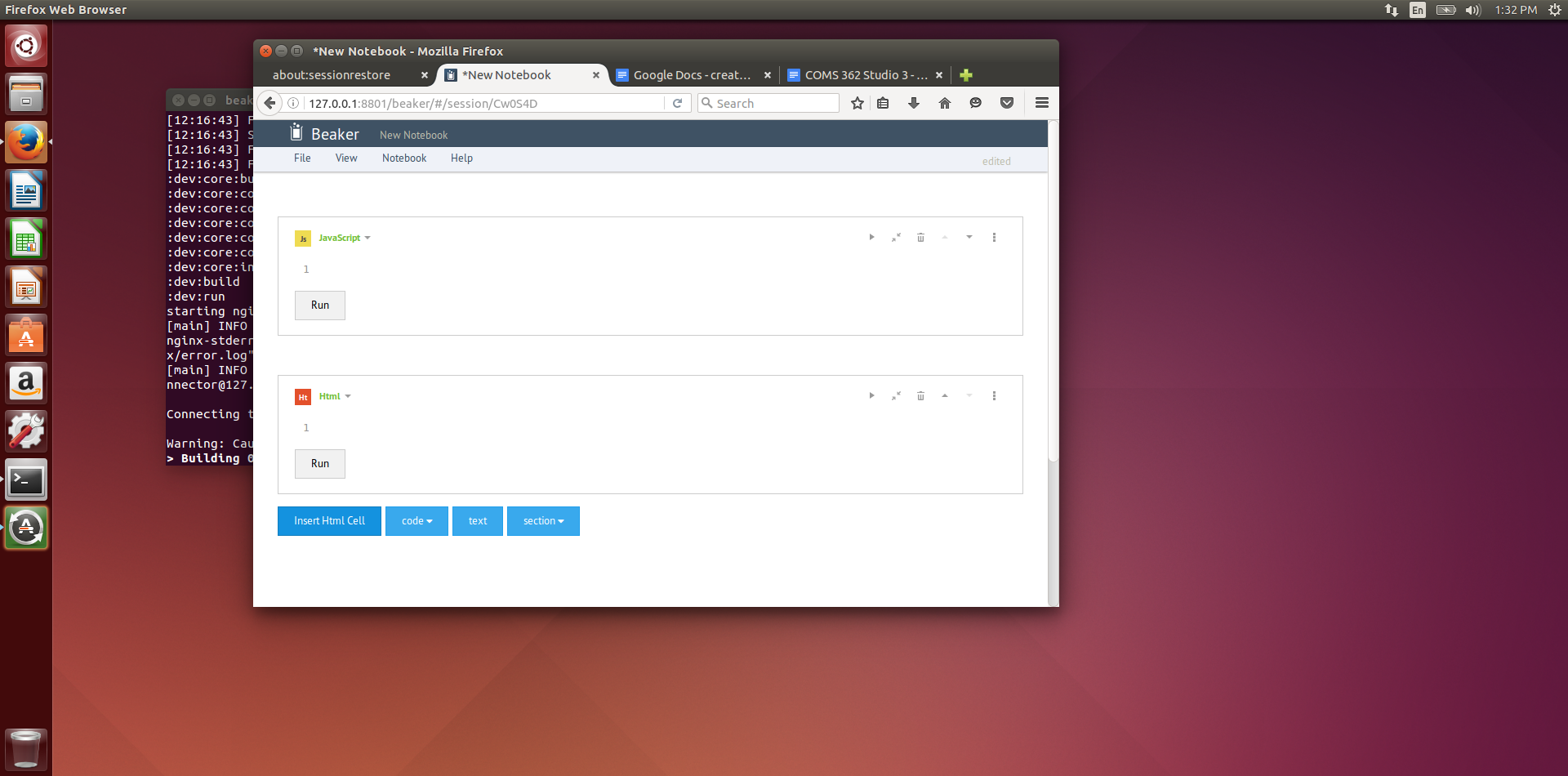
1. The user mouses over the blank area before or after a cell.
2. Beaker displays four options for inserting elements between cells. The first is the option of “Insert <language> Cell”, where <language> is a previously used language in cell of the current notebook.
3. The user wants to create a new cell of the previous language so they select the “Insert <language> Cell” option.
4. Beaker creates a new cell of the specific language type is created under the insert button.

*Alternate Scenario:*

*Trigger Condition:* No cells have been created yet.

*Alternate Scenario Steps:*

1. Beaker displays the options automatically and do not require a mouse over to see them.
2. The “Insert <language> Cell” button is set to the default language, which is currently JavaScript. So Beaker then displays the option “Insert Javascript Cell”.
3. The user wants to start with a Javascript cell, so they select that option.
4. Beaker creates a new cell of the default language type, Javascript, into the empty notebook.



**Use Case 2**

*Use Case Name:* Create New Empty Notebook

*Actor:* User

*Goal:* The user wishes to create a new notebook

*Preconditions:* Beaker application must be launched by the user

*Main Flow:*

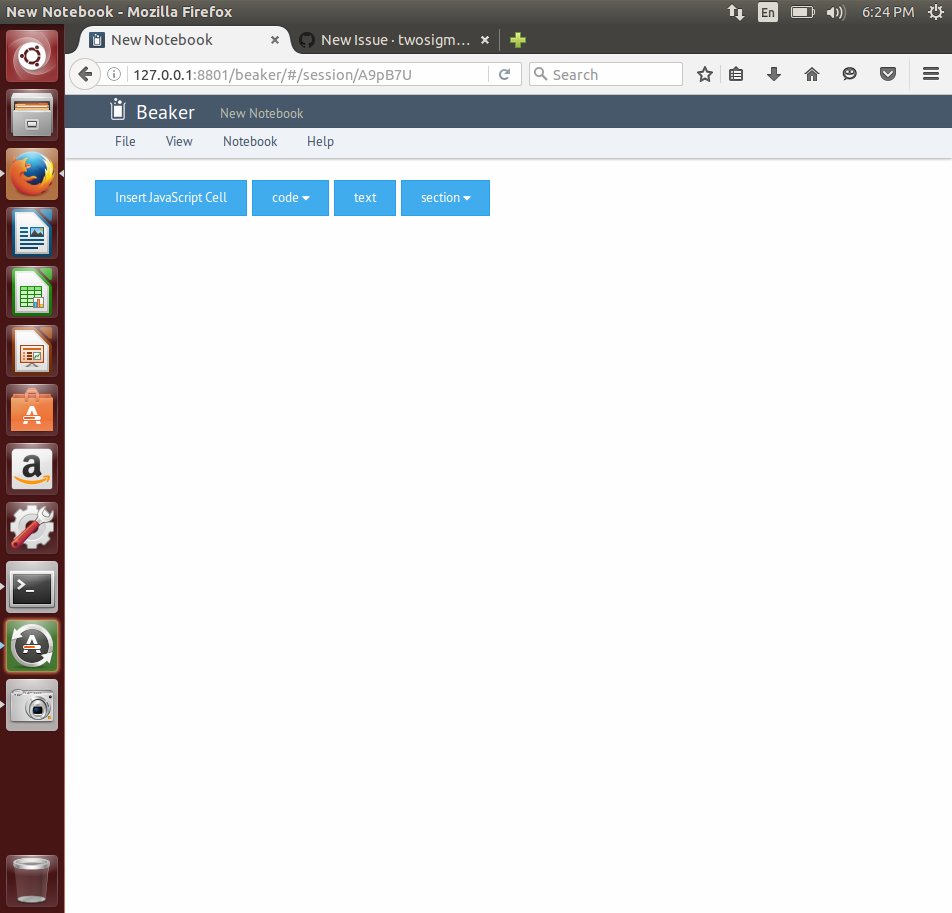
1. Beaker initially displays an interactive interface with two button options for creating a new notebook: create new default notebook and new empty notebook.
2. User wants to create a new empty notebook, so they select the “New Empty Notebook” option.
3. Beaker switches to a new page, with a newly created blank notebook to be edited.

*Alternate Scenario:*

*Trigger Condition:* Beaker initially displays an interactive interface with two button options for creating a new notebook: create new default notebook and new empty notebook.

*Alternate Scenario Steps:*

1. User selects new default notebook.
2. Beaker switches to a notebook named ‘Hello Notebook’ with two, pre-commented cells written in Python, and R.



**Use Case 3**

*Use Case Name:* Report Bug

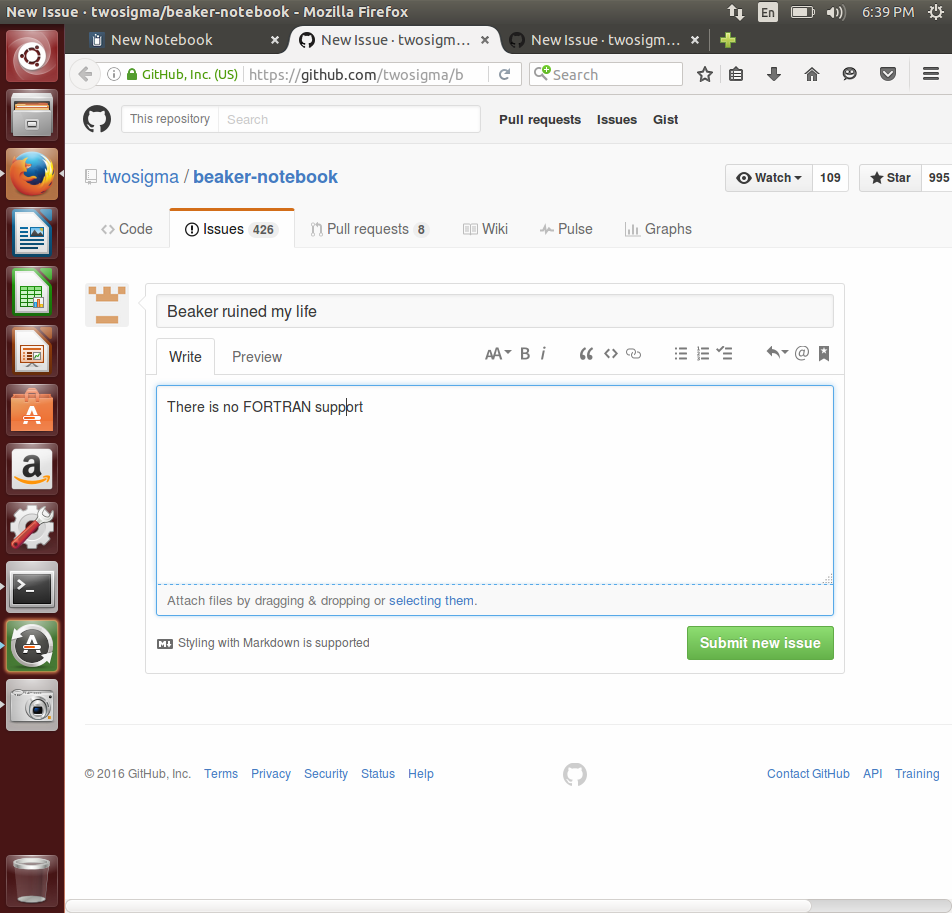
*Actor:* User

*Goal:* The user wishes to report a bug.

*Preconditions:* Beaker application must be launched by the user.

*Main Flow:*

1. Beaker displays a toolbar at all windows, with options of ‘File’, ‘View’, ‘Notebook’, and ‘Help’.
2. User selects ‘Help’.
3. Beaker displays a drop down of clickable options: Tutorial, Keyboard shortcuts, About Beaker, Report a bug or feature request.
4. The user selects Report a bug or feature request.
5. Beaker opens a new tab, redirecting the user to login though an external, source control system - Github.
6. The user enters in username and password.
7. Github authenticates the user, then redirects to the Issues tab of the Beaker notebook repository. Github displays an empty, editable issue document. This document consists of text boxes for a Title and comments, along with a submit button.
8. The user inputs his/her specific issue within the text, and selects the submit button.
9. Github posts this issue onto the repository's Issue tab, and sends a notification to the owners.



**Use Case 4**

*Use Case Name:* Switch to Advanced View

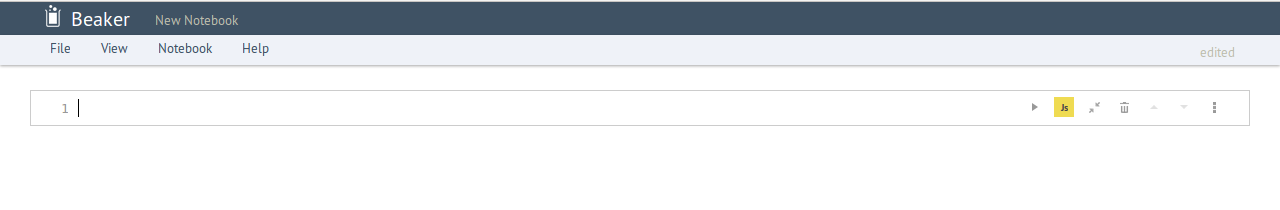
*Actor:* User

*Goal:* The user wishes to switches to a compact view of the notebook.

*Preconditions:* Beaker application must be launched by the user. Also, a notebook must either be newly created, or imported, and the user must be viewing this notebook.

*Main Flow:*

1. Beaker displays a toolbar at all windows, with options of ‘File’, ‘View’, ‘Notebook’, and ‘Help’.
2. User selects ‘View’.
3. Beaker displays a drop down of clickable options: Start Page, Show Hierarchy, Advanced Mode, Show stdout/err.
4. User selects Advanced Mode.
5. Beaker switches to a compact view of the notebook - each notebook cell now has tighter spacing.



**Use Case 5**

*Use Case Name:* New Cell

*Actor:* Users

*Goal:* The user wishes to switch from one language to another.

*Preconditions:* User needs to open Beaker application and then open either an empty notebook or a previously opened one.

*Main Flow:*

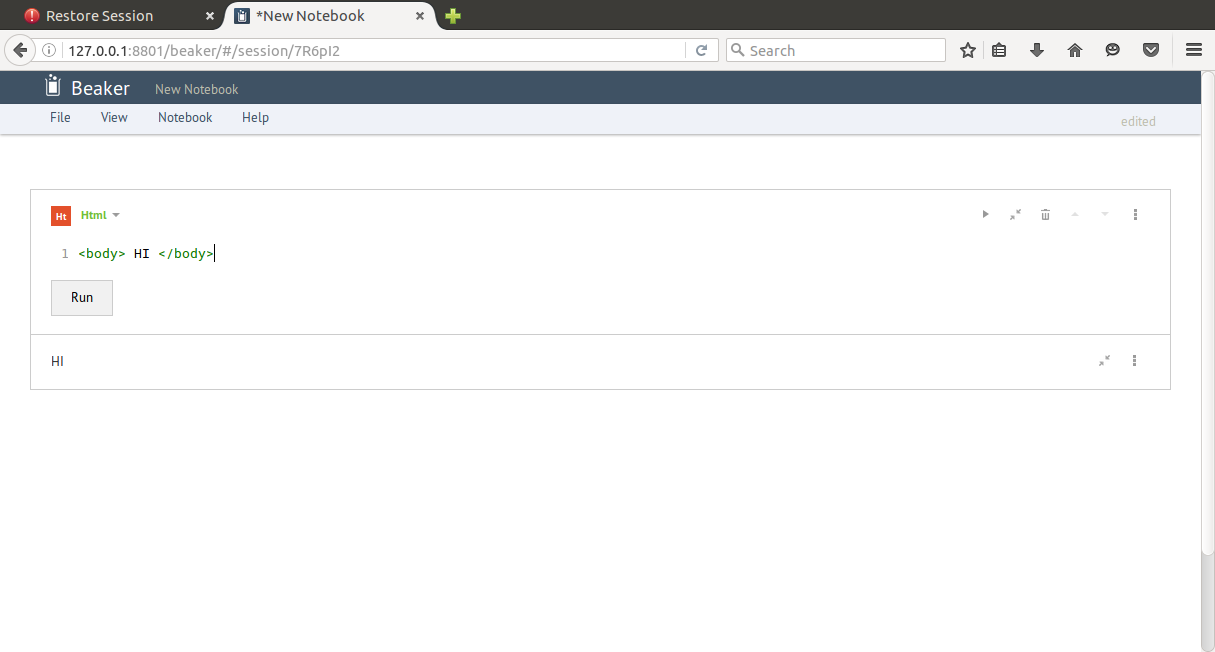
1. Select the dropdown that says, “Code”.
2. Beaker will give the options to select languages, if the user do not see the language that they wish, select the option where it says “Other languages”.
3. Select the radio button for the language the user wish to use.
4. Hit “Close”.
5. Beaker will then give a textbox where the user can enter the code for that particular language.
6. If the user wish to switch the language then user can click on the trash button to delete the textbox and then select the language; follow step #2.

*Alternate Scenario:*

*Trigger Condition:* Beaker usually has two languages that are already pre selected for you, but the user can select any other language that are provided by Beaker.

*Alternate Scenario Steps:*

1. After opening an empty textbox, user can select a any new language by using the down arrow “↓”.



**Use Case 6**

*Use Case Name:* Get a new text box

*Actor:* Users

*Goal:* The user wants to create an empty/new text box.

*Preconditions:* In order to open an empty box the user has to login into Beaker and open an empty notebook so that the user can open the empty textbox.

*Main Flow:*

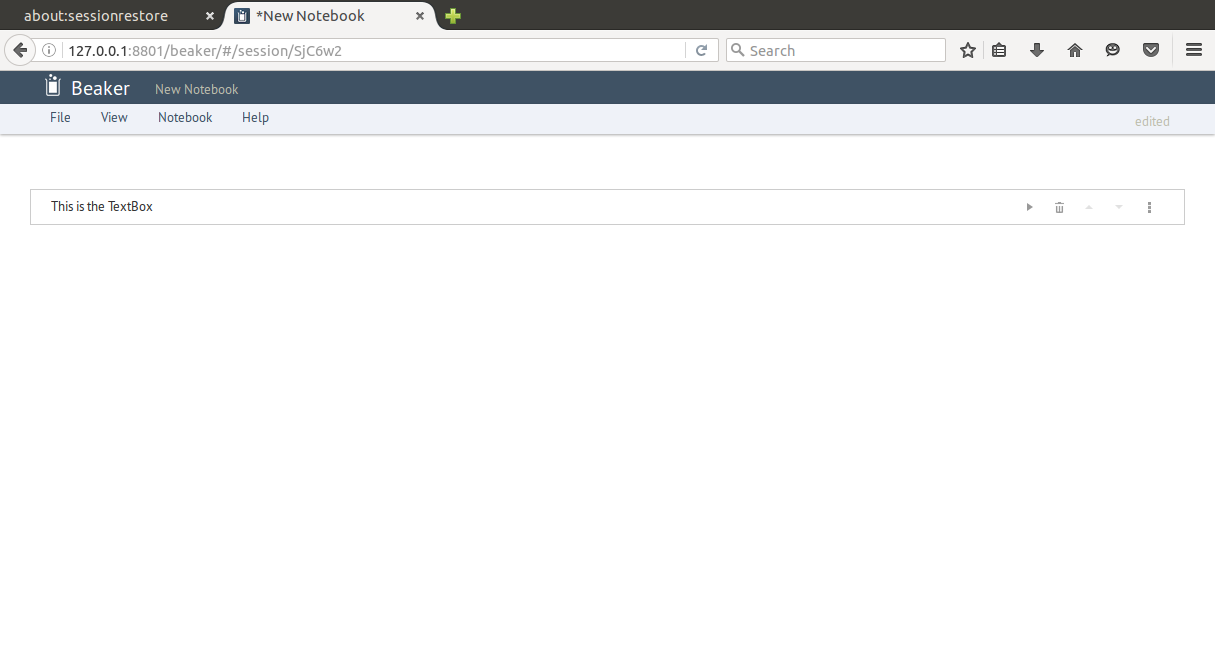
1. After opening an empty notebook, user will be able to see all the cells that are on top of the window, “Language Cell”, “Code”, “Text”, “Section”.
2. In order to get a new text box the user should select text button.
3. After selecting text button, user will be able to type in anything that they wish to.
4. If the user is no longer interested in what was typed, it can be deleted anytime by just clicking trash button located on the right side of the textbox.
5. This textbox can also be runed by clicking “” button located in the textbox right besides where trash button is located.
6. There are also more options that are available that user can use in order to copy and paste.

*Alternate Scenario:*

*Trigger Condition:* Beaker usually has few cells that are available on top of the window screen. One of the option is “text”.

*Alternate Scenario Steps:*

1. Another way you can open the text box is going to “View” dropdown and by selecting textbox.



**Use Case 7**

*Use Case Name:* How to run a program

*Actor:* User

*Goal:* The user wants to run a test after writing a program.

*Preconditions:* In order to run the test, user must write a program in any language that is supported on Beaker.

*Main Flow:*

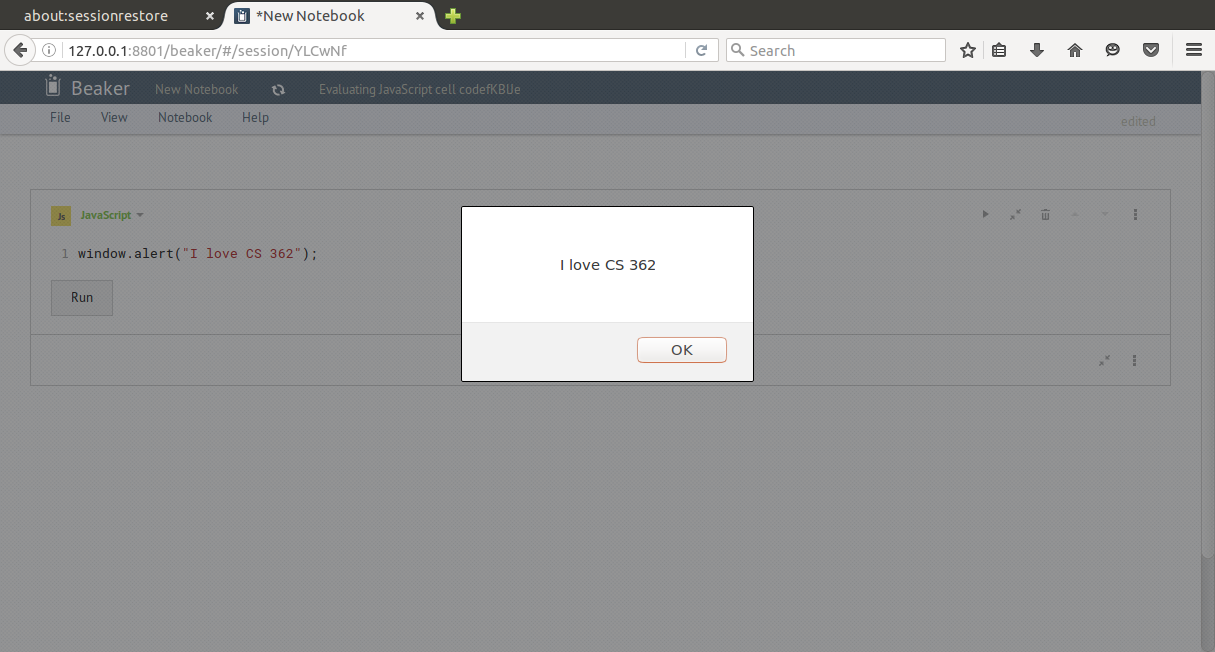
1. User needs to open a notebook in Beaker.
2. User will then see a few options in the toolbar which includes, “Language Cell”, “Code”, “Text”, “Section”.
3. Click on the dropdown option that says “Code”.
4. Select the language that you wish to run the program in.
5. An empty textbox will pop up, the user can type in the code in that textbox.
6. Once the User has completed writing the code, they can click on the Run button which is located right underneath the textbox.
7. After clicking on the button, the user will see the output of the program that was written by the user.
8. If the user wish to switch at any point, there is a dropdown button in the textbox where they can switch to the other language as the user wish to.

*Alternate Scenario:*

*Trigger Condition:* Beaker has a user friendly interface that means the user can switch between screens very easily and can also do simultaneous program at the same time.

*Alternate Scenario Steps:*

1. The first way to run the program is by clicking the “” button which is located underneath the textbox.
2. The second way to run the program is by going under File option which is located at the top of the window, and by selecting “run”.



**Use Case 8**

*Use Case Name:* Open Javascript Tutorial

*Actor:* User

*Goal:* The user wants to find the tutorial manual to read more about using Javascript in Beaker.

*Preconditions:* The user has opened Beaker to its main portal after initially starting the program.

*Main Flow:*

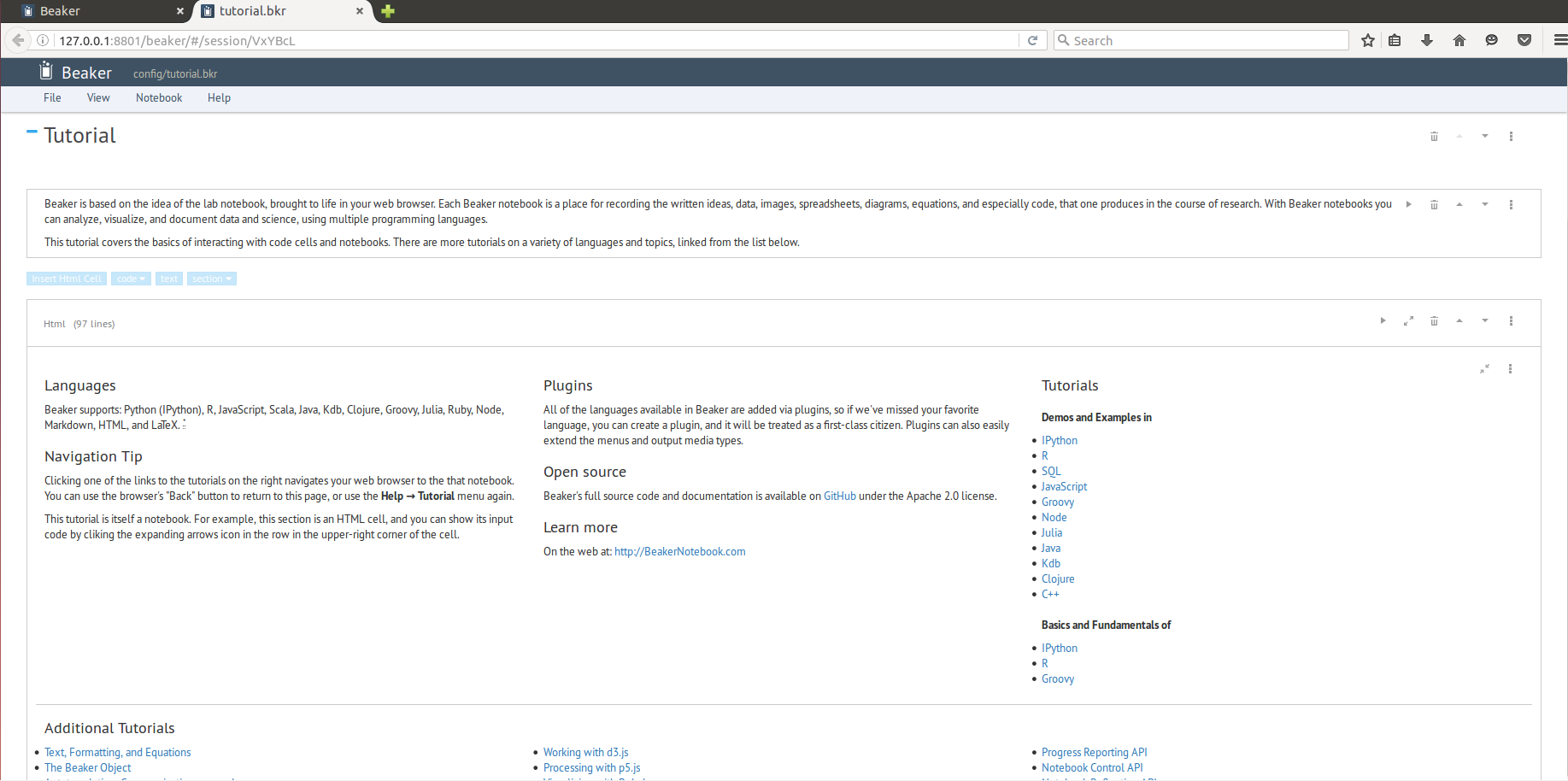
1. The user wants to learn more about Beaker’s Javascript functionality. Across the top of the main portal are several menu options: “File”, “Settings”, and “Help”. Selecting the “Help” menu option displays suboptions, one of which is the “Tutorial” option, which the user selects.
2. Beaker then creates a pre-made notebook called “tutorial.bkr”, that is constructed by the developers in a .bkr notebook format for easy usage within the program. Beaker opens a tab with the notebook open. (Picture below)
3. There are several tutorials for all supported languages, including Javascript. The user sees the Javascript option and selects it.
4. Beaker then opens the specific .bkr project file for demonstrating its support for Javascript functionality. It then displays the project to the user.
5. The user then can run the pre-made samples to show the proper execution, or alter the existing code to do different things. This ensures that all aspects of Javascript are understood and supported by Beaker.
6. The user can then go back to the main Beaker portal, select a different tutorial, or select an option from the drop-down menus.

*Alternate Scenario:* Open Node.js Tutorial

*Trigger Condition:* The user wants to select a different tutorial from Javascript, such as Node.js

*Alternate Scenario Steps:*

1. The user wants to run the Node.js tutorial, so the user selects the “Node” option.
2. Beaker opens the specific .bkr project file for demonstrating Node.js support in Beaker, allowing the user to run several examples.
3. The user can run the pre-made examples to show the execution, or alter the existing code.



**Use Case 9**

*Use Case Name:* Upload file

*Actor:* User

*Goal:* The user uploads a local .bkr file to the Beaker program.

*Preconditions:* The Beaker program is opened to the main page, and the user has a .bkr saved locally on their computer.

*Main Flow:*

1. The user has a local .bkr notebook file on their computer, and they want to edit, run, or save it as another .bkr notebook file. The user selects the “File” menu dropdown at the top, and selects the “Upload (.bkr)” option.
2. Beaker then opens an explorer window to the local file system so the user can browse their hard drive to find a previously saved .bkr notebook file.
3. The user navigates through their file directories until they find the .bkr file they wanted to open, and then user selects it to be uploaded.
4. Beaker opens the file at that location, and either uploads it to the remote server or just opens it locally, depending on how it’s being run by the user. It loads everything required by that notebook and initializes modules for all the necessary languages. Then Beaker displays it to the user.
5. The user can run the code for each cell, or all of them at once, to test out the notebook. From there, the user can either close the notebook, open/upload a new notebook, or return to the main page for Beaker.

*Alternate Scenario 1:*

*Trigger Condition:* The user wants to edit the notebook

*Alternate Scenario Steps:*

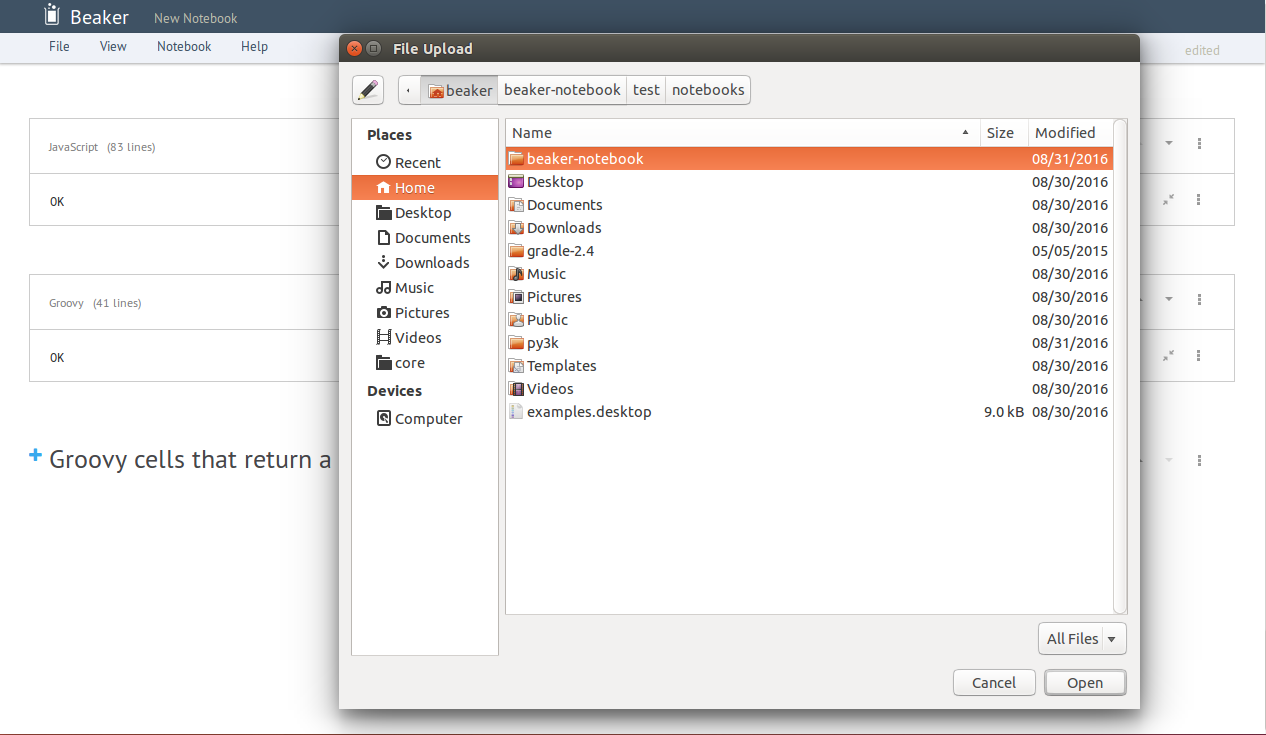
1. Once the notebook has been opened by Beaker, instead of just running the notebook, they can also edit it like any other notebook opened in Beaker. The user makes edits they want and then select the “Save” option in the “File” dropdown menu.
2. Beaker then saves the new edits to the .bkr file on the user’s hard drive to keep the changes after the user has made them.
3. The user can now run the code, edit it more, or close the notebook to accomplish a different task.

*Alternate Scenario 2:*

*Trigger Condition:* The user wants to save the notebook as a new .bkr file.

*Alternate Scenario Steps:*

1. After the notebook has been opened by Beaker and the user has made any edits they wish, the user wants to save the changed notebook by another file name. The user selects the “Save as” option under the “File” dropdown menu.
2. Beaker opens a window for the user to browse through their file system to find the appropriate location to save the file at, and to give the file a given name.
3. The user then finds the desired location, gives the file a name, then selects “Save”.
4. Beaker then saves the current version of the notebook to the given location in the file system with the given filename.
5. The user then can open a new notebook, change the current notebook more, or return to the Beaker home page.



**Use Case 10**

*Use Case Name:* Publish

*Actor:* User

*Goal:* The user wants to publish their notebook so it visible to all Beaker users.

*Preconditions:* A notebook has been created.

*Main Flow:*

1. The user hovers over the File menu.
2. The user clicks the “Publish” button.
3. The user is prompted by the publish dialog to enter a title, description, select a category that their notebook falls into, and to upload an optional picture.
4. The user clicks “Publish”.
5. The user’s notebook is stored within the Beaker database.

*Alternate Scenario 1:*

*Trigger Condition:* The user is not logged into their Beaker account.

*Alternate Scenario Steps:*

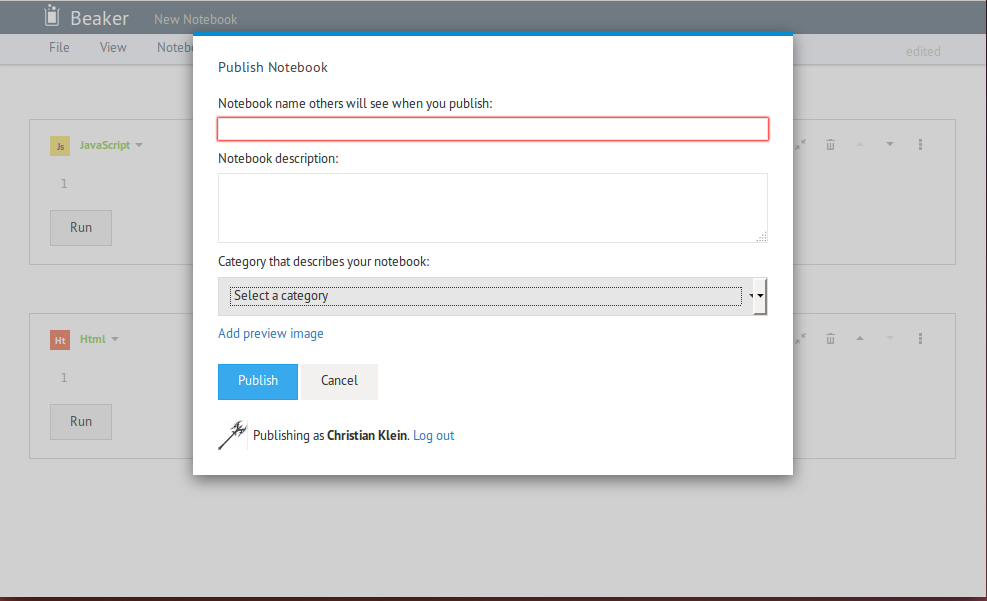
1. When attempting to publish, the user is be prompted by a dialog to enter their Beaker login credentials.
2. The user enters their credentials and logs in.
3. The process resumes from the publish dialog.

*Alternate Scenario 2:*

*Trigger Condition:* The user does not have a Beaker account.

*Alternate Scenario Steps:*

1. When attempting to publish, the user is prompted by a dialog to enter their Beaker login credentials.
2. The user selects “Create an Account” at the bottom of the dialog.
3. The user is redirected to the Beaker website.
4. The user completes the account creation form.
5. A verification email is sent to the email given by the user during account creation.
6. The user verifies the account by accessing their email.
7. The process resumes from the publish dialog.



**Use Case 11**

*Use Case Name:* Delete Cell

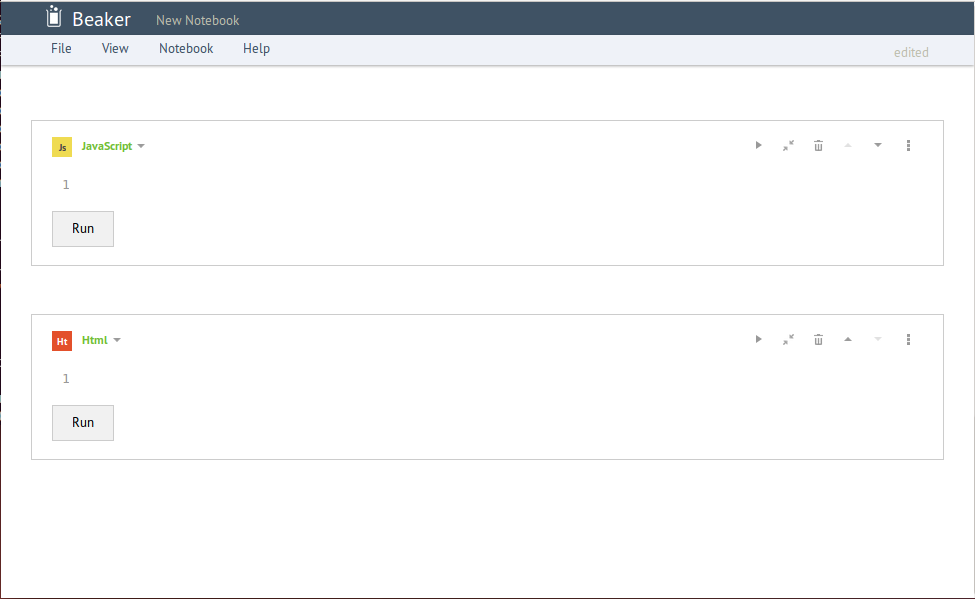
*Actor:* User

*Goal:* The user will delete an undesired cell

*Preconditions:* The user has created a notebook and at least one cell has been instantiated

*Main Flow:*

1. The user wants to delete a cell they have made, so they mouse over the top right corner of the cell that will be deleted and click on the garbage bin icon.
2. Beaker then deletes the selected cell from the current notebook, and surrounding cells are shifted to accommodate for the missing cell.
3. The user can then continue to edit cells, delete cells, or create new cells.



**Use Case 12**

*Use Case Name:* Save Notebook

*Actor:* User

*Goal:* The user want to save his work.

*Preconditions:* The user has created a notebook in Beaker.

*Main Flow:*

1. The user selects the “Save” option from the “File” menu on the top toolbar.
2. If the notebook has not been saved, there will be a new window that appears.
3. The user chooses one sorting mode.
4. The user types the path name into the input field. He can also get the path by choosing the folder from the two file menus.
5. The user adds the file name after the path name in the input field.
6. The user selects “Save” option and saves all cells in the notebook.

*Alternate Scenario 1:*

*Trigger Condition:* The notebook is already saved as a file.

*Alternate Scenario Steps:*

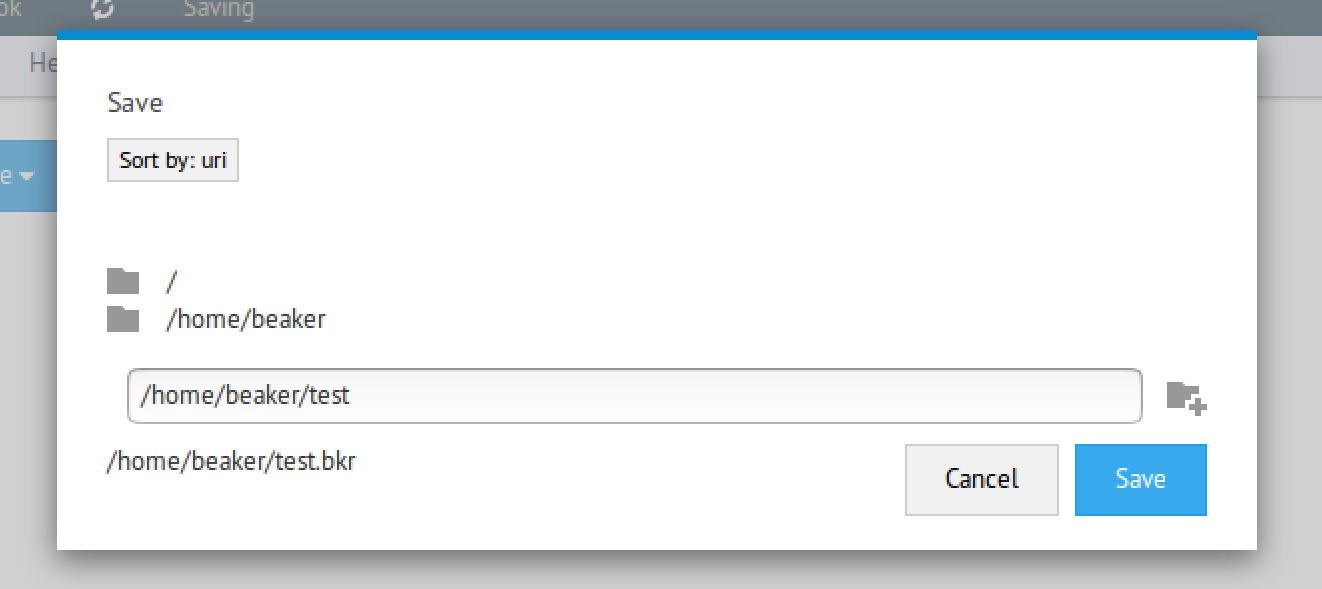
1. The user selects the “Save” option from the “File” menu on the top toolbar.
2. The previous file will be replaced by current notebook.

*Alternate Scenario 2:*

*Trigger Condition:* There is another file that has the same name.

*Alternate Scenario Steps:*

1. The user selects the “Save” option from the “File” menu on the top toolbar.
2. If the notebook has not been saved, there will be a new window that appears.
3. The user chooses one sorting mode.
4. The user types the path name into the input field. He can also get the path by choosing the folder from the two file menus.
5. The user adds the file name after the path name in the input field.
6. The user selects “Save” option.
7. Beaker alerts that the file already exists.
8. The user chooses “Overwrite” option or “Cancel” option.
9. The previous file will be replaced by current notebook if the user chooses “Overwrite” option.



**Use Case 13**

*Use Case Name:* Create sections

*Actor:* User

*Goal:* The user wants to create a section that may contains cells or sections.

*Preconditions:* The user has created and is current opened to a notebook in Beaker.

*Main Flow:*

1. The user puts the mouse between cells or sections at the location where they want to start a new section.
2. Beaker will display four options to the user, each of these are elements that can be inserted between existing cells, or at the beginning or end of the notebook. There is a quick language insertion cell, for example a “Insert Javascript Cell”, a “code” dropdown box for creating a cell of any supported language, a text option for creating an empty text cell, and a section dropdown for separating elements into custom sections.
3. The user wants to create a new section, so they select the “section” option. The user then chooses which level of section they want from the dropdown menu.
4. Beaker then creates a new section of the desired level in the notebook at that location in the notebook..



**Use Case 14**

*Use Case Name:* Run All Cells

*Actor:* User

*Goal:* The User wants to run all the cells.

*Preconditions:* The user has created and opened a notebook in Beaker.

*Main Flow:*

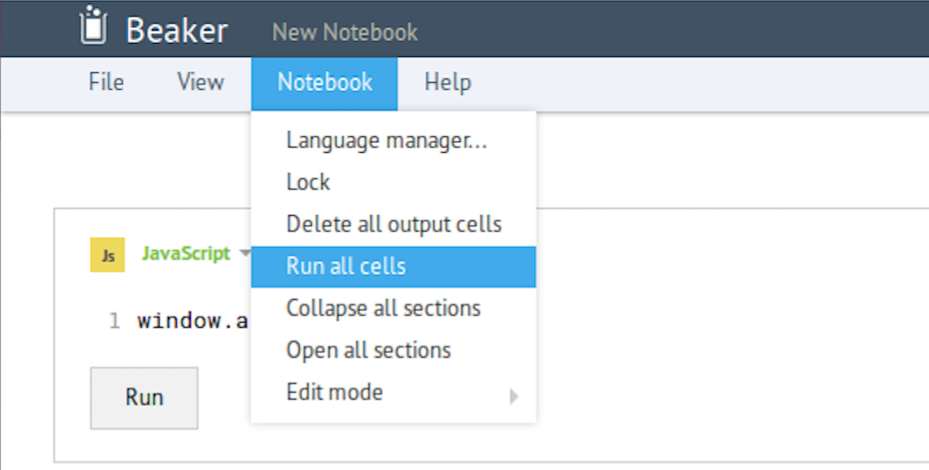
1. The user selects the “Run all cells” option from the “Notebook” menu on the top toolbar.
2. Beaker will run the code in each cell in order that they appear in the notebook. The results of each cell being run will be displayed below the cell that created the results.

*Alternate Scenario:*

*Trigger Condition:* There is an error in one or more cells in the code.

*Alternate Scenario Steps:*

1. The user selects the “Run all cells” option from the “Notebook” menu on the top toolbar.
2. Beaker will run all the cells by their sequence until it stops at the error.
3. There will be an error alert under the specific cell, and all the cells after the error will also be marked as error. Beaker will continue to display errors until all cells can be successfully run in order.



**Com S 362 — Object-Oriented Analysis and Design**

**Certification of Individual Contribution and Understanding Form**

Project Name: Project Use Cases Team Facilitator: Christian Klein

Homework Set: 3 Date: 09/10/2016

Directions: Enter each team member’s name, and a “work rating” (1 - 5 with 5 as high and 1 as low score) that corresponds to the relative share of work done by the team member. Comments to explain work ratings other than 3 must also be provided. The total of the ratings must add up to 3 times the number of team members. Each team member must sign at the end of the form showing your agreement to your team member’s contribution.

*The work rating will have a direct and significant effect on each student’s project and ultimately course grade. In fairness to all students, each team member must give this rating their attention.*

Also, each team member should individually rate the percentage that they understand of the solution. (This does not affect the grade, but is used as a means of communication.)

|  |  |  |  |
| --- | --- | --- | --- |
| Printed team member’s name | Work Rating  (1 - 5) | Comments / Explanations | Understanding Self-Rating  (0 to 100%) |
| Jay Patel | 3 |  | 90% |
| Christian Klein | 3 |  | 90% |
| Joseph Thill | 3 |  | 95% |
| Derek Yu | 3 |  | 90% |
| Zifeng Jiang | 3 |  |  |
| Total: | 15 |  | |

Note that you must make Total = 3 \* (Number of team members)

|  |
| --- |
| I agree to the above ratings and understand our team’s solution. (Each team member should sign their name below.) |
| Christian Klein |
| Jay Patel |
| Derek Yu |
| Zifeng Jiang |
| Joseph Thill |