Case 1: Load the page	
Use Case:	User types the url of the plagiarism checker and sees the homepage of the checker.
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	Allow the user to access the plagiarism checker.
Preconditions:	NA
Trigger:	User presses the ‘Enter/ Return’ key.
Scenario:	1. User types the url of the plagiarism checker. 2. User press the ‘Enter/ Return’ key. 3. The homepage for the checker is loaded.
Exceptions:	The homepage is not loaded successfully.
When available:	First Increment
Chanel to actor:	Web browser

Case 2: Upload the files	
Use Case:	User selects a single file or recursively selects a folder and containing files and uploads the assignment submission to the plagiarism checker.
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	To check whether the assignment submission is plagiarized.
Preconditions:	User is on the homepage of the checker.
Trigger:	User clicks the ‘Upload’ button.
Scenario:	1. User gets access to the homepage of the plagiarism checker. 2. User chooses to upload the assignment submission. 3. User selects the files. 4. User confirms to upload the chosen files.
Exceptions:	Files might be too large for the plagiarism checker. User might need to upload the second time if fails at the first time.
When available:	First Increment
Chanel to actor:	UI of the plagiarism checker providing the ‘Upload’ button

Case 3: Drag and drop the files	
Use Case:	User drags and drops the assignment submission to the plagiarism checker.
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	To check whether the assignment submission is plagiarized.
Preconditions:	User is on the homepage of the plagiarism checker.
Trigger:	User drops files in the required section.
Scenario:	1. User gets access to the homepage of the plagiarism checker. 2. User selects the files. 3. User drags the selected files. 4. User drops the selected files in the required section.
Exceptions:	Files might be too large for the plagiarism checker. User might need to drag and drop the files a second time if fails at the first time.
When available:	Second Increment
Chanel to actor:	UI of the plagiarism checker providing a ‘Drag and Drop to upload’ label

Case 4: Start to compare the files	
Use Case:	User chooses to start the check.
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	To check whether the chosen assignment submission is plagiarized.
Preconditions:	User is on the homepage of the plagiarism checker and the comparing files have been uploaded or dropped in the plagiarism checker.
Trigger:	The ‘Compare’ button is clicked.
Scenario:	1. User gets access to the homepage of the checker. 2. User finishes uploading or dropping the selected files on both panels of the checker. 3. User clicks the ‘Compare’ button.
Exceptions:	The plagiarism checker fails to respond.
When available:	First Increment
Chanel to actor:	UI of the plagiarism checker providing a ‘Compare’ button

Case 5: Navigate across the similar lines of code	
Use Case:	User navigates line by line to check the highlighted similar lines of code.
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	Manually check suspicious code to see if the program is plagiarized.
Preconditions:	User is on the homepage of the plagiarism checker. The comparing files have been uploaded or dropped in the checker.User clicks the ‘Compare’ button. The checker has finished detection.
Trigger:	User clicks ‘next’/ ‘previous’ button.
Scenario:	1. User gets access to the homepage of the checker. 2. User finishes uploading or dropping the selected files on both panels of the checker. 3. User clicks the ‘Compare’ button. 4. If any two pieces of code are similar, the corresponding lines of code should be highlighted and displayed on both panels of the plagiarism checker. 5. By clicking ‘next’/ ‘previous’ button, user can navigate to next or previous similar lines of code.
Exceptions:	The plagiarism checker fails to respond.
When available:	Second Increment
Chanel to actor:	UI of the plagiarism checker providing the ‘next’ / ‘previous’ button

<table>
<tr><th colspan="2">Case 6: Reset the comparison</th></tr>
<tr><td><b>Use Case:</b></td><td>User finishes the current check and begins another attempt to check for plagiarism.</td></tr>
<tr><td><b>Primary Actor:</b></td><td>User (Anonymous user, logged in user)</td></tr>
<tr><td><b>Goal in Context:</b></td><td>Enable user to start a new plagiarism checker.</td></tr>
<tr><td><b>Preconditions:</b></td><td>User is on the homepage of the plagiarism checker. The comparing files have been uploaded or dropped in the checker. User clicks the ‘compare’ button. The plagiarism checker has finished detection.</td></tr>
<tr><td><b>Trigger:</b></td><td>User clicks ‘reset’ button.</td></tr>
<tr><td><b>Scenario:</b></td><td>1. User gets access to the homepage of the plagiarism checker.
2. User finishes uploading or dropping the selected files on both panels of the checker.
3. User clicks the ‘Compare’ button.
4. The plagiarism checker has finished the check.
5. After user clicks the ‘Start new comparison’ button, the current panels will be reset and user can start to upload or drop new files to start a new check for plagiarism.</td></tr>
<tr><td><b>Exceptions:</b></td><td>The plagiarism checker fails to respond.</td></tr>
<tr><td><b>When available:</b></td><td>First Increment</td></tr>
<tr><td><b>Chanel to actor:</b></td><td>1. UI of the checker providing the ‘reset’ button
2. A popup alert window reminding if the user does want to start a new comparison.</td></tr>
</table>

Case 7: Send feedback	
Use Case:	User gives feedback on the plagiarism checker.
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	Help the team to build a better plagiarism checker.
Preconditions:	User is on the homepage of the checker.
Trigger:	User clicks ‘Feedback’ button.
Scenario:	1. User gets access to the homepage of the checker. 2. User presses the ‘Feedback’ button. 3. User can add feedback in the popup window.
Exceptions:	The popup fails to generate.
When available:	Third Increment
Chanel to actor:	1. UI of the plagiarism checker providing a ‘Feedback’ button 2. a popup feedback window

Case 8: User chooses to play the demo video tutorial	
Use Case:	User chooses to play the demo video tutorial after loading the website
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	To enable the user to view the demo video
Preconditions:	User should be on the homepage of the checker and shouldn’t close the demo video which plays automatically
Trigger:	User has loaded the website and hasn’t closed the demo video that plays automatically
Scenario:	User loads the homepage of the plagiarism checker.
Exceptions:	User’s internet speed might be low for a video to play. User might not have the required plug-in to watch the video

Case 9: User chooses to create an account	
Use Case:	User chooses to create an account.
Primary Actor:	Anonymous user
Goal in Context:	To enable the user to create an account on the plagiarism checker
Preconditions:	User should be on the homepage of the plagiarism checker.
Trigger:	User clicks on the ‘Sign Up’ button
Scenario:	1. User clicks on the ‘Sign Up’ button 2. User types in the required details 3. User clicks on the ‘Submit’ button 4. User is redirected to the ‘Sign in’ page
Exceptions:	1. User might enter different values in ‘Password’ and ‘Retype Password’ field – In this case, type again 2. User might not enter some field which is necessary to fill in – In this case, type again 3. The plagiarism checker fails to respond.
Chanel to actor:	UI of the plagiarism checker providing a ‘Sign Up’ button

Case 10: User want to exclude files	
Use Case:	User wants to exclude some files from the project that has been added for checking
Primary Actor:	User (Anonymous user, logged in user)
Goal in Context:	To exclude the files which are not to be checked for plagiarism
Preconditions:	User is on the homepage of the plagiarism checker and the comparing files have been uploaded or dropped in the plagiarism checker.
Trigger:	The ‘Advanced options’ button is clicked.
Scenario:	1. User drops the files that are to be compared in the plagiarism checker 2. User clicks on ‘Advanced options’ 3. User enters the files names which are to be excluded 4. User clicks the ‘Done button.
Exceptions:	1. Files names entered are not present in the uploaded project – Type file names again in this scenario 2. The plagiarism checker fails to respond.
Chanel to actor:	UI of the plagiarism checker providing an ‘Advanced options’ button

Case 11: User saves a plagiarism report	
Use Case:	User saves a plagiarism report
Primary Actor:	User (Logged in user)
Goal in Context:	Enable the user to save the plagiarism report in the checker which can be viewed later under ‘History’
Preconditions:	User should have an account and should be logged in. User should have ran a successful plagiarism check
Trigger:	User clicks the ‘Save Report’ button.
Scenario:	1. User has run a successful plagiarism check. 2. User clicks the ‘Save Report button. 3. The report is saved and can be viewed later
Exceptions:	1. User isn’t logged in. 2. User hasn’t run a plagiarism check 3. The plagiarism checker fails to respond.
Chanel to actor:	UI of the plagiarism checker providing the ‘Save Report’ button

<table>
<tr><th colspan="2">Case 12: User chooses to view history</th></tr>
<tr><td><b>Use Case:</b></td><td>User chooses to view the history of his/her plagiarism checks</td></tr>
<tr><td><b>Primary Actor:</b></td><td>Logged in user</td></tr>
<tr><td><b>Goal in Context:</b></td><td>Enable user to check the history of plagiarism checks performed by him/her</td></tr>
<tr><td><b>Preconditions:</b></td><td>User should have an account and should be logged in</td></tr>
<tr><td><b>Trigger:</b></td><td>User clicks the ‘History’ button.</td></tr>
<tr><td><b>Scenario:</b></td><td>1. User clicks on the ‘User Name’ drop down
2. User clicks on the ‘History’ button
3. User is able to check the history of his/her checks</td></tr>
<tr><td><b>Exceptions:</b></td><td>The plagiarism checker fails to respond.</td></tr>
<tr><td><b>Chanel to actor:</b></td><td>UI of the checker providing the ‘History’ button</td></tr>
</table>

Case 13: Sign In	
Use Case:	User clicks on the ‘Sign In’ button
Primary Actor:	Anonymous user
Goal in Context:	Enable the user to log in to the plagiarism checker
Preconditions:	User should not be logged into the plagiarism checker
Trigger:	User clicks the ‘Sign In’ button.
Scenario:	1. User clicks on the ‘Sign In’ drop down 2. User enters his/her email id and password 3. User clicks on the ‘Log In’ button 4. User is logged in to the plagiarism checker
Exceptions:	The plagiarism checker fails to respond.
Chanel to actor:	UI of the plagiarism checker providing a ‘Sign In’ button

Case 14: Sign out	
Use Case:	User clicks on the ‘Sign Out’ button
Primary Actor:	Logged in user
Goal in Context:	Enable the user to log out of the plagiarism checker
Preconditions:	User should be logged into the plagiarism checker
Trigger:	User clicks the ‘Sign Out’ button.
Scenario:	1. User clicks on the ‘User Name’ drop down 2. User clicks on the ‘Sign Out’ button 3. User is logged out of the plagiarism checker
Exceptions:	The plagiarism checker fails to respond.
Chanel to actor:	UI of the plagiarism checker providing a ‘Sign Out’ button