

Sprint 3 Manual Testing Documentation

Team 21: Jialu Gu, Shafer Hess, Emily Ou, Derek Shu, Mangkorn Yuan

This document will outline the procedures and results of manually testing some of the user stories as specified in our sprint planning document. Any user story tests not mentioned in this document were tested with unit tests or test scripts.

User Story 2 Test:

User Story: "As a user, I would like to have access to a tutorial so that I can learn my way around the app when I first download it."

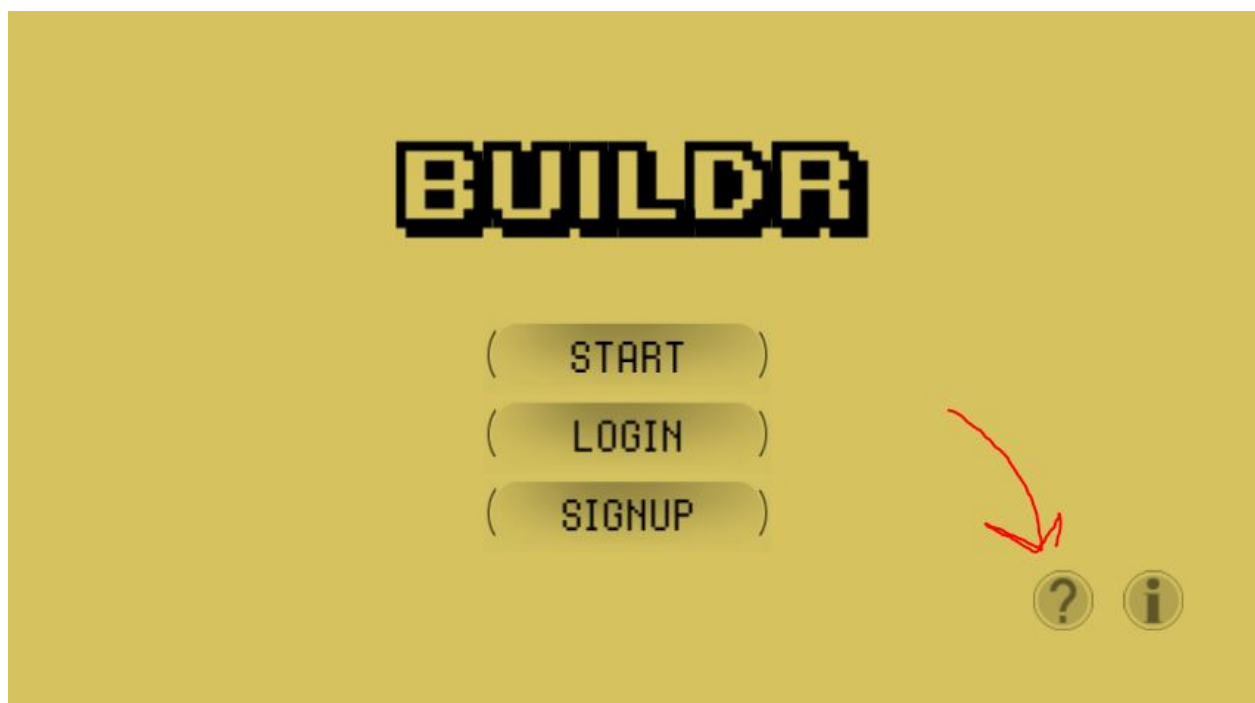
Test: Tutorial is accurate in terms of how to maneuver the app

Testing Procedure:

1. Navigate to Tutorial page
2. Read the tutorial and apply the steps given to see if tutorial is accurate (expected to be) - Testing View Creation
 - a. Press Start AR
 - b. Aim phone camera at detection carpet
 - c. Creation should appear as expected

Results:

1. To get to the Tutorial page was successful by clicking the bottom right '?' icon from the starting menu prior to login.



2. The tutorial text is easy to read.



3. Following the tutorial, Pressing Start AR is functional.



-

Testing Summary: The manual testing for User Story 2 was successful. Clicking the '?' icon accesses the tutorial where I could scroll down and read the tutorial. It was accurate as I was able to get to AR mode with no issues and setting my camera on the detection carpet was successful. It allowed a creation to pop up in the AR world.

User Story 3 Test:

User Story: “As a user, I would like to be able to enable or disable being publically visible in the AR world in case I want to hide from others in the AR world.”

Test: Test that uploaded models are properly tagged as private or public in backend

Testing Procedure:

1. Tag a model as public
2. Tag a model as private
3. Go to admin page to see if that model is tagged as private

Results:

1. The “Make Your Creation Private?” check box allows a user to
 - a. Check the box if they want to make it private
 - b. Leave box empty if the user wants to make their creation public

The screenshot displays a yellow-themed 'UPLOAD' interface. At the top, the word 'UPLOAD' is written in a large, pixelated font. Below it, the interface is divided into three main sections: 1. PICK A BUILDING, 2. PICK A NAME, and 3. UPLOAD!. In the first section, a dropdown menu shows 'LAWSON' with a downward arrow. Below this, instructions read: 'TO VIEW YOUR CREATIONS AFTER UPLOADING, HEAD TO THE SELECTED PURDUE BUILDING, FIND THE DETECTION CARPET, TURN ON AR MODE, AND AIM YOUR PHONE CAMERA TOWARDS THE CARPET.' In the second section, a button labeled 'COOLBUILDING' is shown, followed by a checkbox and the text 'MAKE YOUR CREATION PRIVATE?'. In the third section, a 'DONE' button is visible. To the right of the 'DONE' button, there is a 'CURRENT DETECTION CARPET:' label and a small image of a building with 'LAWSON' written on it. At the bottom of the screen, a 'BACK' button is present.

UPLOAD

1. PICK A BUILDING

LAWSON ▼

TO VIEW YOUR CREATIONS AFTER UPLOADING, HEAD TO THE SELECTED PURDUE BUILDING, FIND THE DETECTION CARPET, TURN ON AR MODE, AND AIM YOUR PHONE CAMERA TOWARDS THE CARPET.

2. PICK A NAME

COOLBUILDING

☐ MAKE YOUR CREATION PRIVATE?

3. UPLOAD!

(DONE)

(BACK)

CURRENT DETECTION CARPET:



2. Models are updated as public or private respectively in the database as intended.
 - a. Models tagged as “1” are private
 - b. Models tagged as “0” are public

The screenshot shows the Buildr Admin Dashboard with the 'Models' tab selected. The 'All Models' section displays a table of model entries. The 'Privacy' column is highlighted with a red box, indicating that models with a value of '1' are private and models with a value of '0' are public. The table includes columns for Model ID, Build Data, Build Name, Username, Location, Privacy, Avatar, and Upvotes.

Model ID	Build Data	Build Name	Username	Location	Privacy	Avatar	Upvotes
3	{shipName!default...}	build3	sampleName3	sampleLocation3	1	0	1
4	{shipName!default...}	***** replace*****	sampleName3	sampleLocation3	0	1	0
5	{shipName!default...}	***** replace*****	sampleName3	sampleLocation3	0	1	0
6	{shipName!default...}	aaa	sampleName3	sampleLocation3	0	1	0
7	{shipName!default...}	****2	sampleName3	sampleLocation3	0	1	0
8	{shipName!default...}	****pack	sampleName3	sampleLocation3	0	0	0
9	{shipName!default...}	****pack	sampleName3	sampleLocation3	0	0	0
10	{shipName!default...}		sampleName3	sampleLocation3	0	0	0
11	{shipName!default...}		sampleName3	sampleLocation3	0	0	0
13	{shipName!default...}	crossy	sampleName3	sampleLocation3	0	0	0

Showing 1 to 10 of 27 entries

Previous 1 2 3 Next

Refresh Models

Testing Summary: The manual testing for User Story 3 was successful. Tagging Model Name “build3” as private and uploading that model was saved in the database as private. It can be seen in the admin Models table that private models are tagged as “1” under the privacy column and public models are “0”.

User Story 5 Test:

User Story: "As a user, I would like to be able to view the change log for updates to the app so I can see what features have been added or removed for each build."

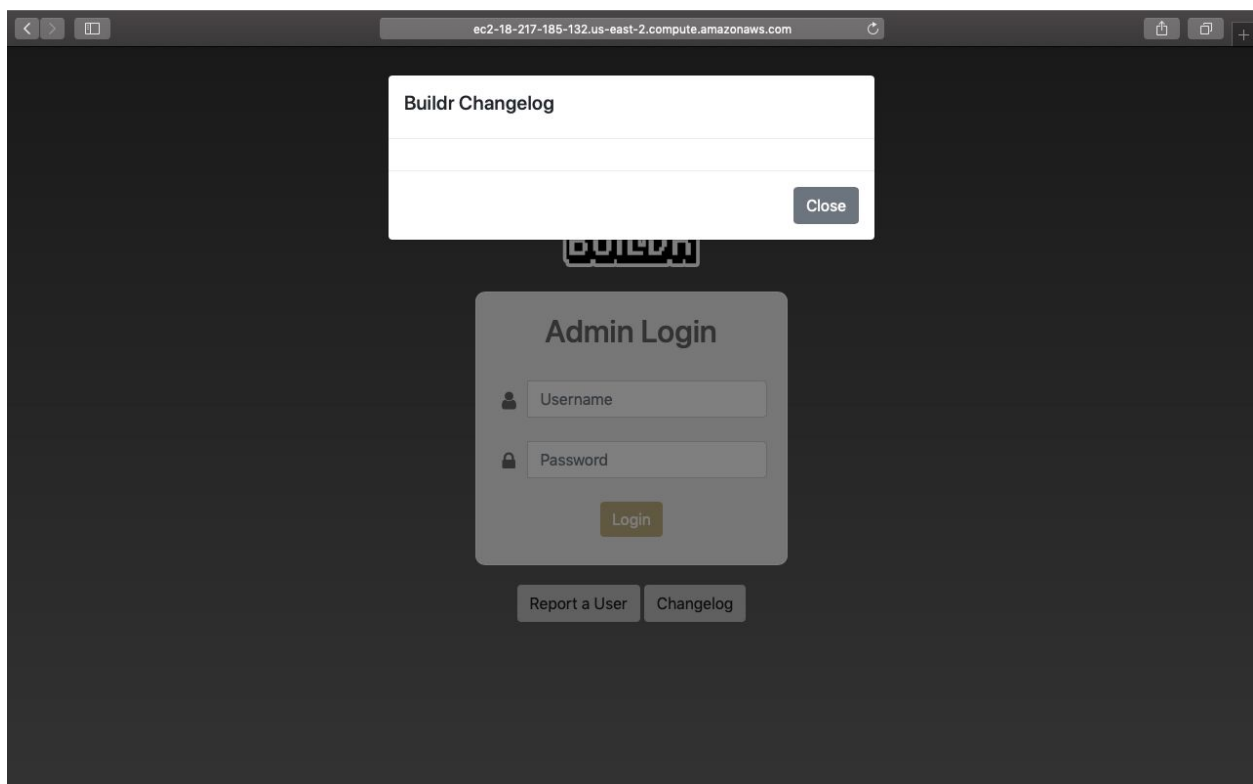
Test: Verify that new changelogs are receivable and readable from the client

Testing Procedure:

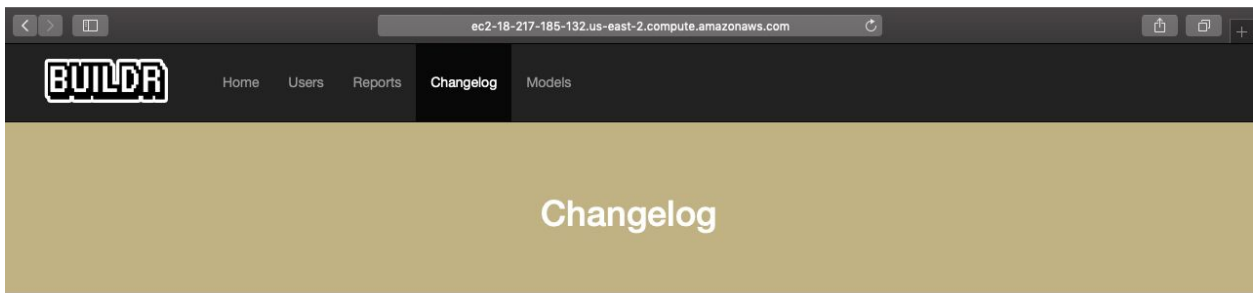
1. Click Changelog (should be empty for testing) on admin website
2. Login as admin and go to changelog page
3. Edit the changelog (add text)
4. Submit the changelog
5. Refresh
6. Go to login page and click changelog - compare results to expected (expected-changelog is updated)

Results:

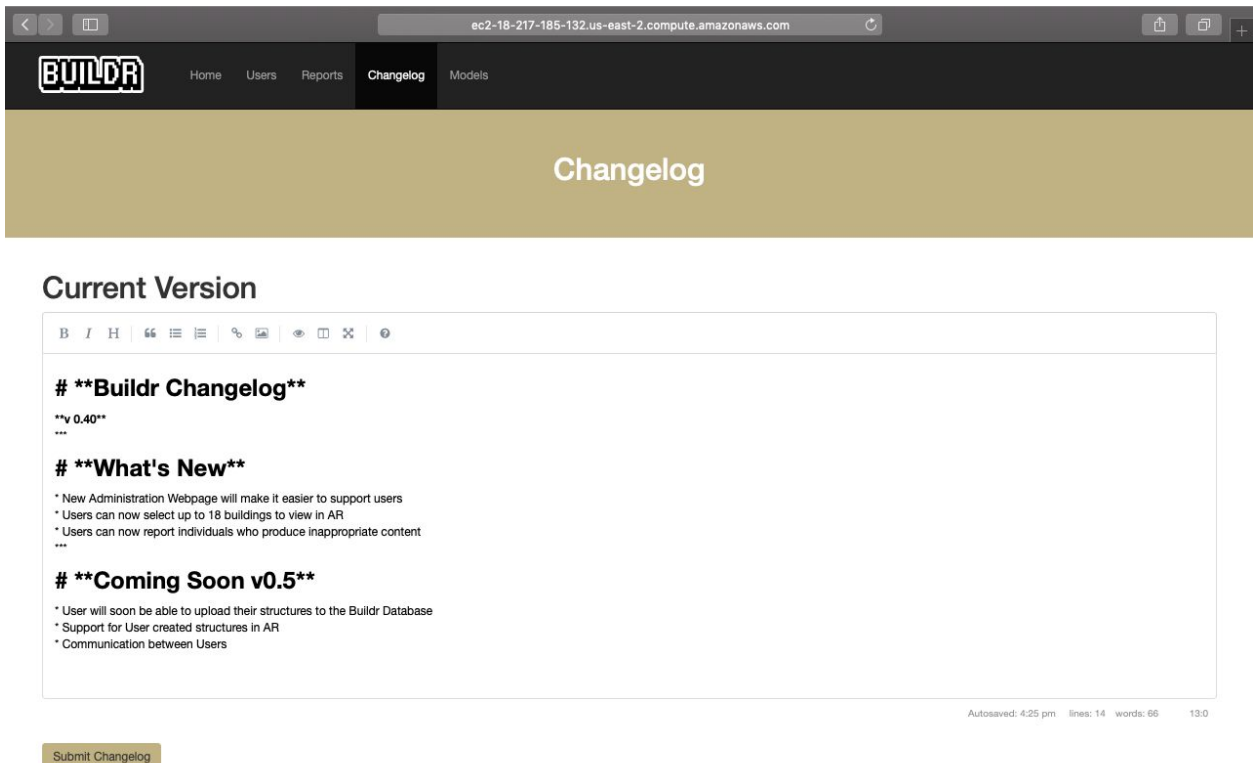
1. Changelog button works and changelog is empty for testing purposes.



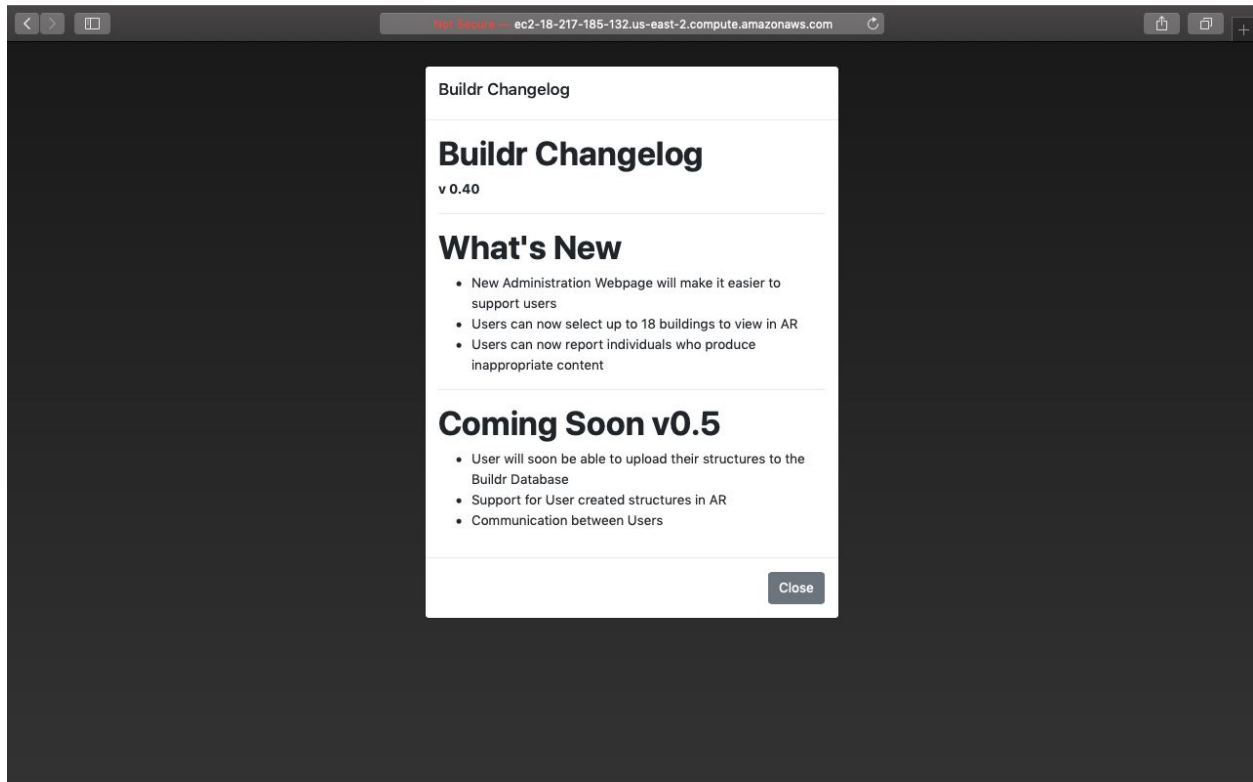
2. Logged in as administrator and navigating to changelog page was successful.



3. Adding text to the changelog is successful as text can added without issues.



4. After adding the text, submitting the new changelog and refreshing the page, going back to the login page you can see the changelog was successfully changed.



Testing Summary: The manual testing for User Story 5 was successful. The changelog button to view the changelog is functional on the login page and navigating to changelog page on the administrator website is also functional. Adding text works without any issues and submitting works too. After submitting and refreshing the changelog page we can view on the login page that the changelog was indeed received and readable from the client.

User Story 6 Test:

User Story: "As a user, I would like to be able to friend other users on the platform so that I can communicate with them through direct messages."

Test: Verify that a user can add other users as friends (by username) and direct message them inside the interface

Testing Procedure:

*Note the User for this testing procedure is 1u

1. Click Social to get to the DM/add friends screen
2. Add friends (2u & 3u) by username in the "To add friend" textbox (expected to add)
3. Select a friend to message
4. Send messages between two users (expected to send)
5. Switch to message another friend to prove direct messages works for all friends - should send without issues

Results:

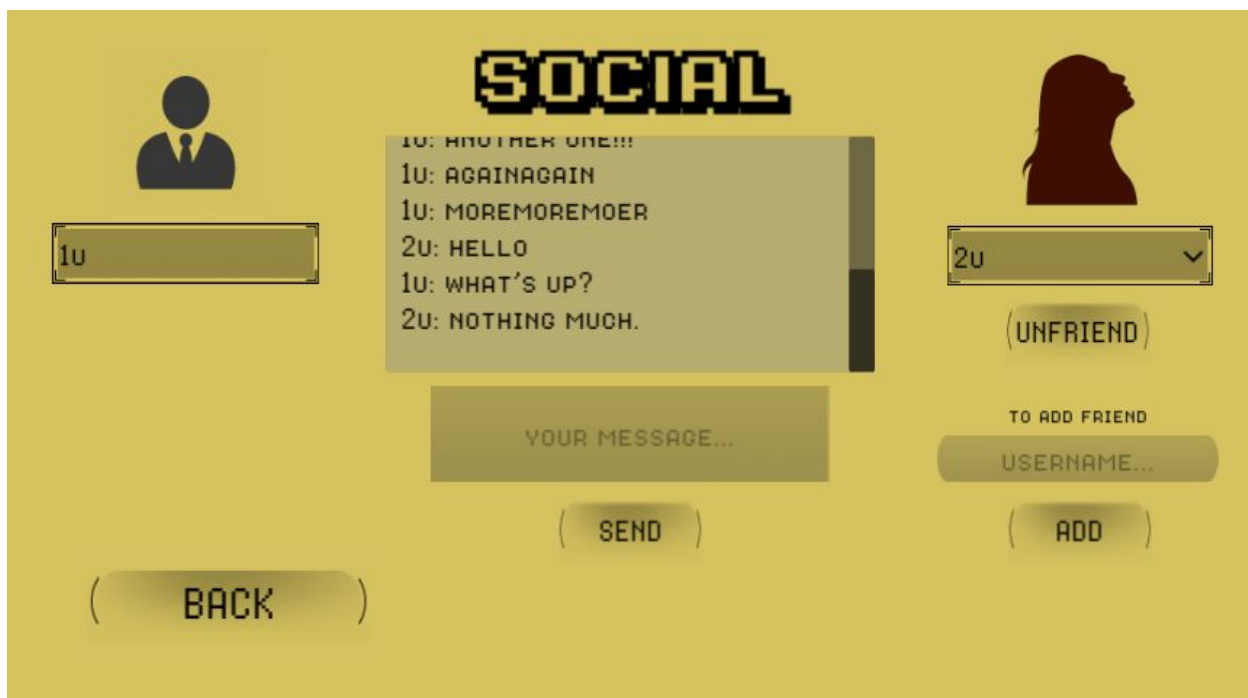
1. Clicking Social leads to the Social screen was successful.



2. Adding friend by username was successful. Both friends user "2u" and "3u" were added successfully.



3. Selecting a friend to send direct messages with is functional and the direct messages were also sent without issues.



4. Switching between friends also works as messages can be received from all friends.



Testing Summary: The manual testing for User Story 6 was successful. Users can add friends using the "To Add Friend" textbox and add the user they want to friend by username. Messages can be sent between friends without issues. And the user can add multiple friends and send direct messages between these friends.

User Story 7 Test:

User Story: “As an administrator, I would like to be able to ban (or timeout) other users from the platform to deal with troublemaking users.”

Test: (1) Verify that an administrator can punish other players through the control panel and that the punishment is recorded in the database and properly restricts the users actions

Testing Procedure:

1. Navigate to admin Users page
2. Ban user with the username “user” by typing the username under the “Delete User” textbox
3. Press Delete
4. Press Refresh Users (expected - user to be removed from Users table on admin page)

Results:

1. Navigating to the User’s page on administrator website was successful. Table of users is shown below.

Website Not Secure

BUILDOR Home **Users** Reports Changelog Models Log Out

Users

All Users

Show 10 entries

First Name	Last Name	Email	Username	Rank	Status	University
asdf	asd	asd	asd	default	good	purdue
Barry	Manilow	Manilow@gmail.com	ManilowBarry	default	good	purdue
bob	marley	urthfjf	newbobby	default	good	purdue
Derek	Shu	hess33@purdue.edu	teaspoon	default	good	purdue
djdjdjd	sjdjd	angkom.yuan99@gmail.com	manghgh	default	good	purdue
first	last	email	user	default	good	purdue
Mangkorn	Yuan	mangkorn.yuan08@gmail.com	mang	default	good	purdue
Model	User	User@gmail.com	UserModel	default	good	purdue
Pekoe	TiaBar	pekotbar@gmail.com	pekotbar	default	good	purdue
Shafer	Hess	shu17@purdue.edu	grr	default	good	purdue
First Name	Last Name	Email	Username	Rank	Status	University

Showing 1 to 10 of 12 entries

Previous 1 2 Next

Refresh Users

Delete User

Enter Username:

Delete

Update Status

Enter Username:

Good

Update

Update Rank

Enter Username:

Admin

Update

2. Typing the user that is to be banned into the Delete User was successful. There was not any issues with it.

The screenshot shows a web application interface for managing users. At the top, there's a navigation bar with 'BUILDOR' and links for Home, Users, Reports, Changelog, and Models. A 'Log Out' button is in the top right. Below the navigation bar is a header section titled 'Users'. The main content area is titled 'All Users' and displays a table of 12 users. The table has columns for First Name, Last Name, Email, Username, Rank, Status, and University. The user 'user' is highlighted with a red border. Below the table, there are three buttons: 'Refresh Users', 'Delete User', 'Update Status', and 'Update Rank'. A blue arrow points from the 'Delete User' button to a modal window titled 'Delete User'. The modal contains the text 'Enter Username:' and a text input field with the value 'user'.

First Name	Last Name	Email	Username	Rank	Status	University
asdf	asdf	asdf	asdf	default	good	purdue
Barry	Manlowe	Manlowe@gmail.com	ManloweBarry	default	good	purdue
bob	manley	ucf@cf	newbobdy	default	good	purdue
Derek	Shu	hesa338@purdue.edu	tespoon	default	good	purdue
dididid	eddyd	anghorn-yue08@gmail.com	mangygn	default	good	purdue
first	last	email	user	default	good	purdue
Mangkhorn	Yuan	mangkhorn.yuan08@gmail.com	mang	default	good	purdue
Model	User	User@gmail.com	UserModel	default	good	purdue
Pekoe	Tiedlar	pekoe@bar@gmail.com	pekoe@bar	default	good	purdue
Shafar	Hena	shu178@purdue.edu	grt	default	good	purdue

Showing 1 to 10 of 12 entries

Previous 1 2 Next

Refresh Users

Delete User

Enter Username:

User

Delete

Update Status

Enter Username:

Good 2

Update

Update Rank

Enter Username:

Admin 2

Update

Delete User

Enter Username:

user

3. After pressing Delete and Refresh Users, the user was successfully deleted from the database and no longer shown on the admin Users Table.

The screenshot shows the BUILDOR application interface. At the top, there's a navigation bar with 'Home', 'Users', 'Reports', 'Changelog', and 'Models'. The 'Users' section is active. Below the navigation bar, the title 'Users' is displayed. The main content area shows 'All Users' with a table containing 12 entries. The table has columns: First Name, Last Name, Email, Username, Rank, Status, and University. The user 'User' is highlighted in red. Below the table, there's a 'Refresh Users' button. A blue arrow points to this button with the label 'Refresh Users Button'.

First Name	Last Name	Email	Username	Rank	Status	University
asd	asd	asd	asd	default	good	purdue
Barry	Manlow	Manlow@gmail.com	ManlowBarry	default	good	purdue
bob	marley	urjhjf	newbobby	default	good	purdue
Derek	Shu	hess33@purdue.edu	teaspoon	default	good	purdue
djdjdj	ejdjdj	angkom.yuan98@gmail.com	mangghg	default	good	purdue
First	last	email	user	default	good	purdue
Mangkom	Yuan	mangkom.yuan98@gmail.com	mang	default	good	purdue
Model	User	User@gmail.com	UserModel	default	good	purdue
Pekoe	TeaBar	pekotbar@gmail.com	pekotbar	default	good	purdue
Shafer	Hess	shu17@purdue.edu	grr	default	good	purdue
First Name	Last Name	Email	Username	Rank	Status	University

Showing 1 to 12 of 12 entries

Previous 1 2 Next

Refresh Users

The screenshot shows the 'Delete User' form. It has a text input field for 'Enter Username:' with the value 'user'. Below the input field is a 'Delete' button. A green arrow points to this button with the label 'Delete Button'.

Delete User

Enter Username: user

Delete

The screenshot shows the BUILDOR application interface. At the top, there's a navigation bar with 'Home', 'Users', 'Reports', 'Changelog', and 'Models'. The 'Users' section is active. Below the navigation bar, the title 'Users' is displayed. The main content area shows 'All Users' with a table containing 11 entries. The user 'User' is no longer present. Below the table, there's a 'Refresh Users' button. A blue arrow points to this button with the label 'Refresh Users Button'.

First Name	Last Name	Email	Username	Rank	Status	University
asd	asd	asd	asd	default	good	purdue
Barry	Manlow	Manlow@gmail.com	ManlowBarry	default	good	purdue
bob	marley	urjhjf	newbobby	default	good	purdue
Derek	Shu	hess33@purdue.edu	teaspoon	default	good	purdue
djdjdj	ejdjdj	angkom.yuan98@gmail.com	mangghg	default	good	purdue
Mangkom	Yuan	mangkom.yuan98@gmail.com	mang	default	good	purdue
Model	User	User@gmail.com	UserModel	default	good	purdue
Pekoe	TeaBar	pekotbar@gmail.com	pekotbar	default	good	purdue
Shafer	Hess	shu17@purdue.edu	grr	default	good	purdue
Shafer	Hess	shafer.hess01@gmail.com	hess33	administrator	good	purdue
First Name	Last Name	Email	Username	Rank	Status	University

Showing 1 to 10 of 11 entries

Previous 1 2 Next

Refresh Users

The screenshot shows the 'Delete User' form. It has a text input field for 'Enter Username:' with the value 'user'. Below the input field is a 'Delete' button. The 'Delete' button is now disabled.

Delete User

Enter Username: user

Delete

Testing Summary: The manual testing for User Story 7 was successful. The administrator can access the User Table on its website and enter the username it wishes to delete due to a violation. After pressing Delete then Refresh Users the user selected (user) was deleted from the database and no longer shown on the admins Users Table.

User Story 8 Test:

User Story: "As an administrator, I would like to be able to censor inappropriate words in the comments and replace them with stars."

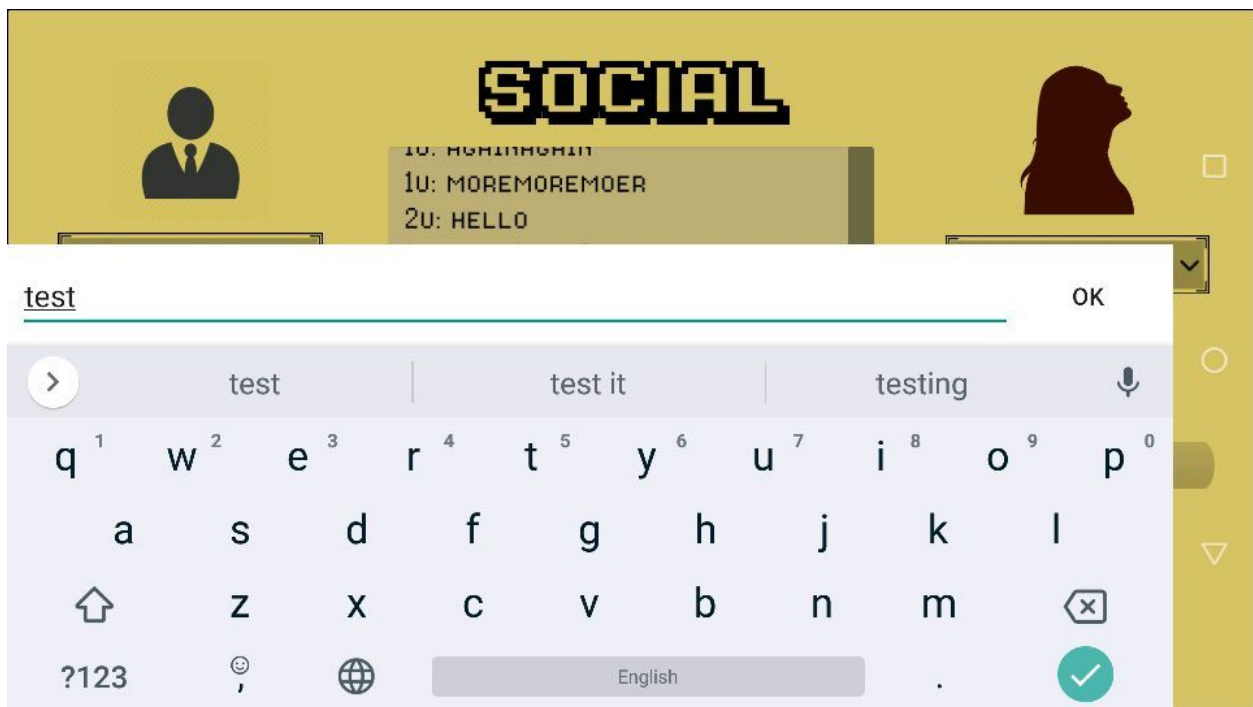
Test: Inappropriate words are censored to all users in the chat

Testing Procedure:

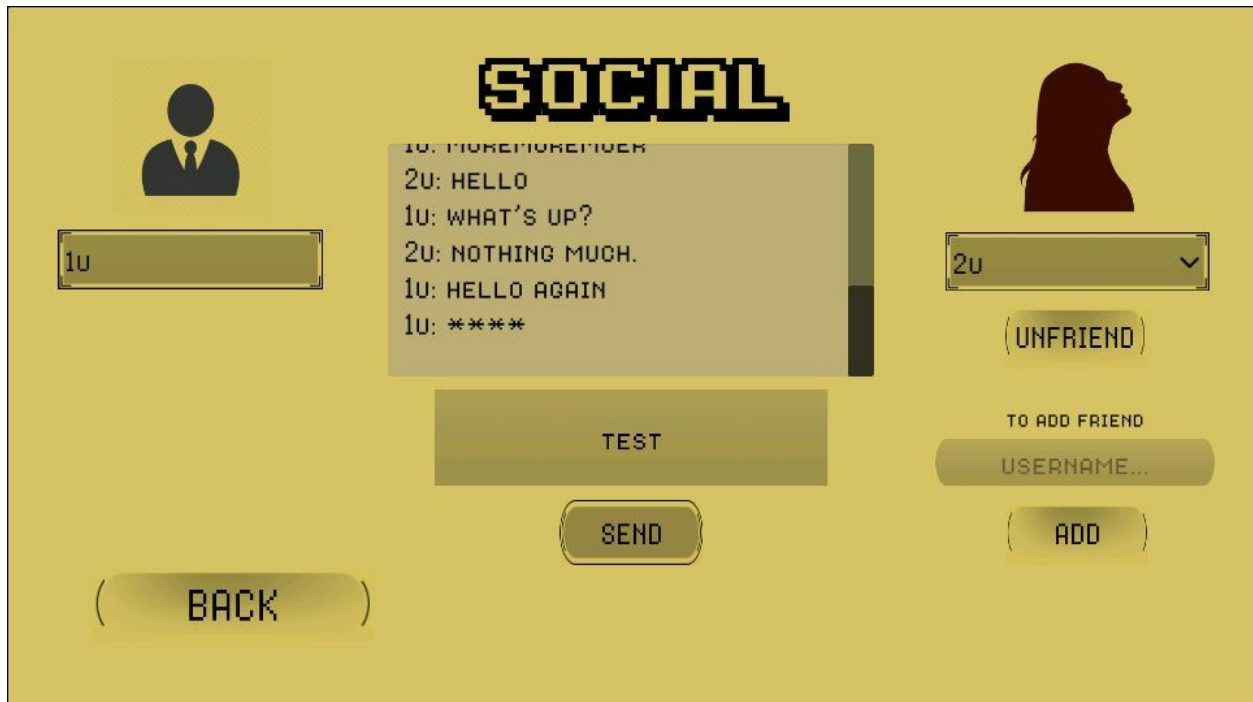
1. Type a message containing and inappropriate word (the word "test" is censored for this testing)
2. Send the message and compare results to expected (expected to be replaced with '*'s)

Results:

1. The chat is functional. "Test" (our censored word) could be typed in the user's text box.



2. After pressing send the inappropriate word "test" was successfully replaced with '*'s.



Testing Summary: The manual testing for User Story 8 was successful. After using the word "test" as our censored word for this test, it was successfully replaced with '*'s as expected to.

User Story 9 Test:

User Story: "As a user, I would like to be able to report inappropriate content made by other users so that the content will be reviewed and possibly removed."

Test: Users can report and administrator decides to remove content if terms are violated, admins can decide to remove

Testing Procedure:

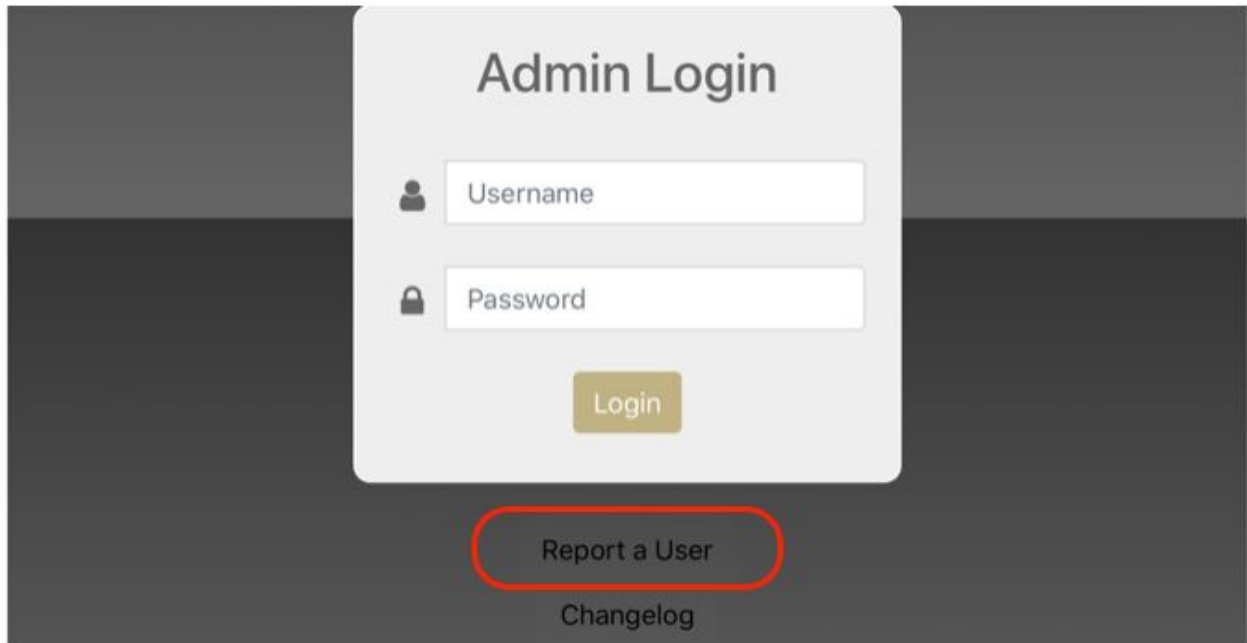
1. Click the mail button in the signup page
2. Click Report a User
3. Enter the username to report (UserModel) reason for report (Goes to IU)
4. Click Report User
5. Login and click refresh reports
6. Check if report is added to the admin reports table (expected)
7. If not violated - release report and no further actions is needed
 - a. Enter Report ID (7) that I wish to remove
 - b. Press Release then Refresh Reports (report expected to be gone)
8. If violated - remove the content
 - a. *See User Story 12*
 - b. Repeat the release report steps

Results:

1. User report button is functional and sends the user to the admin site where the user can report the user.

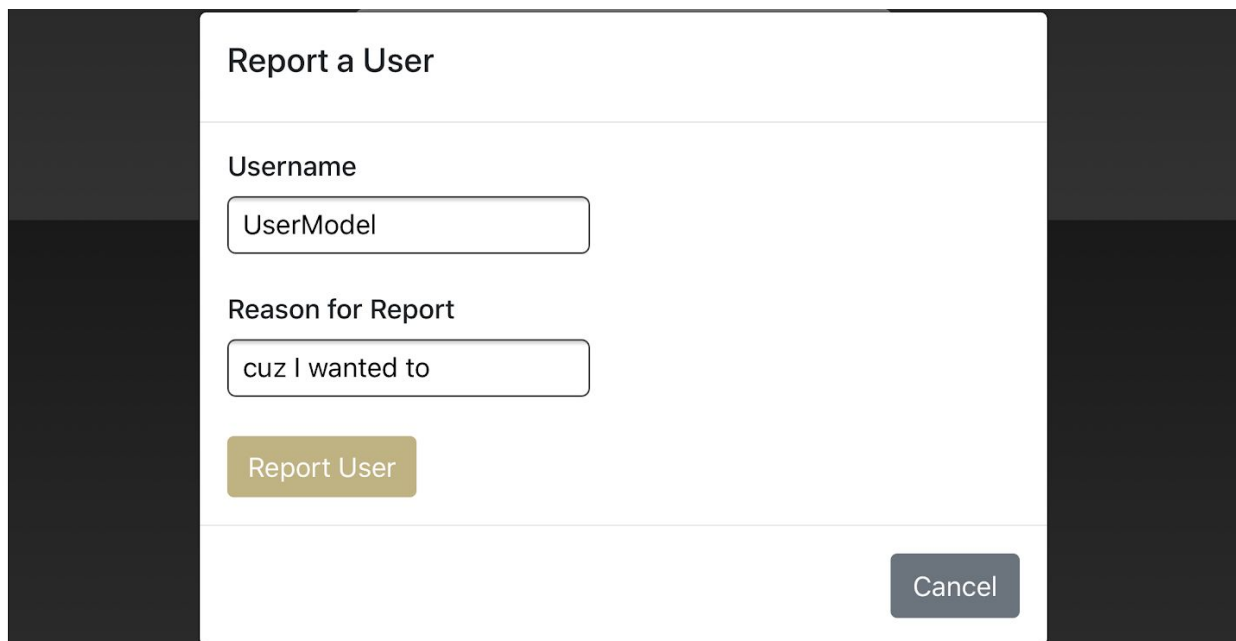


2. Clicking Report a User is functional and by doing so the UI to enter the information pops up.



The image shows a dark-themed user interface for an 'Admin Login' system. A light gray modal box is centered, containing a title 'Admin Login', a 'Username' input field with a user icon, a 'Password' input field with a lock icon, and a 'Login' button. Below this modal, a 'Report a User' button is highlighted with a red rounded rectangle. A 'Changelog' link is visible at the bottom center.

3. Username and reason for report can be entered and submitted by pressing "Report User" successfully.



The image shows a 'Report a User' modal form. It has a title bar 'Report a User'. Below the title, there are two input fields: 'Username' with the value 'UserModel' and 'Reason for Report' with the value 'cuz I wanted to'. At the bottom left is a 'Report User' button, and at the bottom right is a 'Cancel' button.

4. After pressing "Refresh Reports" report is successfully added to the admin reports table.

The screenshot shows the BUILDA application interface. The top navigation bar includes 'Home', 'Users', 'Reports' (active), 'Changelog', and 'Models'. A 'Log Out' button is in the top right. Below the navigation bar is a 'Reports' section header. The main content area is titled 'Received Reports' and features a table with 5 entries. The table has columns for 'Report ID', 'Username', and 'Report Text'. The 7th entry is highlighted with a red box. Below the table, there is a 'Refresh Reports' button, which is also highlighted with a red box and labeled 'Refresh Reports Button' with a red arrow. Below the button is a 'Release Report' section with a form to 'Enter Report ID' and a 'Release' button.

Received Reports

Show 10 entries Search:

Report ID	Username	Report Text
1	shafer	terrible coder
2	derek	best SQL coder
4	shaferTest	reported for bad coding
5	samplename	sampletext
7	UserModel	cuz I wanted to

Showing 1 to 5 of 5 entries Previous 1 Next

Refresh Reports ← Refresh Reports Button

Release Report

Enter Report ID

Release

- After deciding there is no violation then report can be released successfully after entering the Report ID then pressing Release then Refresh Reports. Report is released and no longer included in the admin reports table.

Received Reports

Show entries

Report ID	Username	Report Text
1	shafer	terrible coder
2	derek	best SQL coder
4	shaferTest	reported for bad coding
5	samplename	sampletext
7	UserModel	cuz I wanted to

Showing 1 to 5 of 5 entries

Previous Next

Refresh Reports

Release Report
Enter Report ID

Release Report
Enter Report ID

Received Reports

Show entries

Report ID	Username	Report Text
1	shafer	terrible coder
2	derek	best SQL coder
4	shaferTest	reported for bad coding
5	samplename	sampletext

Showing 1 to 4 of 4 entries

Previous Next

Refresh Reports

Release Report
Enter Report ID

6. The results of a violation and steps taken to remove the violation was successful and shown in User Story 12.

Testing Summary: The manual testing for User Story 9 was successful. Clicking the mail icon led to the admin login page where the user can click "Report a User" and enter the respective fields. After pressing send, the report is sent to the database where the admin can view the report after clicking "Refresh Reports". If an admin decides that there was no violation with the report than the admin can release the report and no further action is needed. If the admin decides that there was a violation then the admin can proceed with the steps described in User Story 12.

User Story 10 Test:

User Story: “As a user, I would like to be able to have full control and transparency over my privacy settings and data collection so I know exactly what data is being collected and what data I’m providing.”

Test: Spell check, grammar check, and properly shows in app

Testing Procedure:

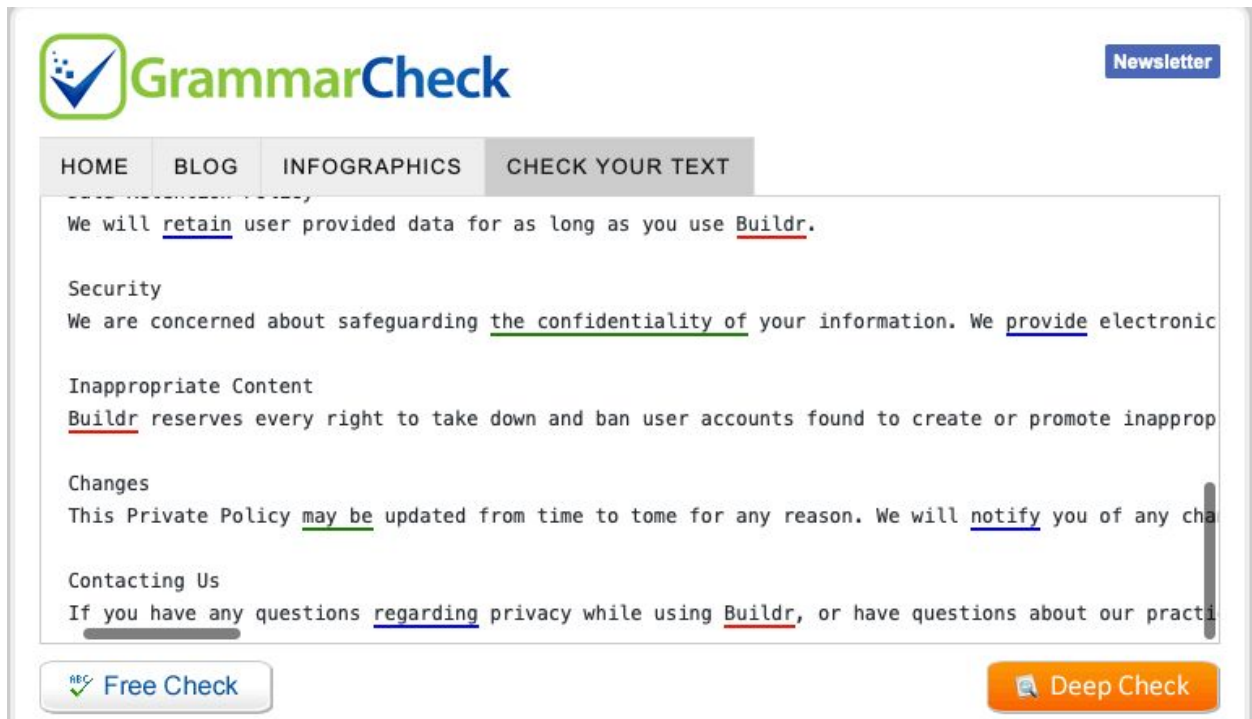
1. Put the privacy policy text in a spell check website
2. Correct the spelling mistakes if any
3. Put the privacy policy text in a grammar check website
4. Correct the grammar mistakes if necessary
5. Check privacy policy on app (expected to show without any issues)

Results:

1. Privacy Policy text in spell check website www.jspell.com and corrected any spelling errors detected by the website

Not Found encourges	Replace	Not Found officilas	Replace
Replace With encourage	Replace All	Replace With officials	Replace All
Suggestions encourage encourages encourager encounter	Ignore	Suggestions officials official official's	Ignore
	Ignore All		Ignore All
	Learn		Learn
	Finish		Finish

2. Privacy Policy text in grammar check website www.grammarcheck.net and corrected any grammar errors detected by the website (green underline is the suggested grammar mistakes)



The screenshot shows the GrammarCheck website interface. At the top left is the GrammarCheck logo. To the right is a 'Newsletter' button. Below the logo is a navigation bar with links: HOME, BLOG, INFOGRAPHICS, and CHECK YOUR TEXT. The main content area displays a privacy policy text with several grammar suggestions highlighted by green underlines. The text includes sections for 'Security', 'Inappropriate Content', 'Changes', and 'Contacting Us'. At the bottom of the page, there are two buttons: 'Free Check' and 'Deep Check'.

GrammarCheck

Newsletter

HOME BLOG INFOGRAPHICS CHECK YOUR TEXT

We will retain user provided data for as long as you use Buildr.

Security

We are concerned about safeguarding the confidentiality of your information. We provide electronic

Inappropriate Content

Buildr reserves every right to take down and ban user accounts found to create or promote inappropri

Changes

This Private Policy may be updated from time to tome for any reason. We will notify you of any cha

Contacting Us

If you have any questions regarding privacy while using Buildr, or have questions about our practi

Free Check Deep Check

3. The Privacy Policy successfully shows on the app under Credits with no issues. The entire Policy can be seen using the scrollbar.



The screenshot shows the 'CREDITS' screen of the Buildr app. The screen has a yellow background. On the left side, there is a list of credits for the project, including the names of the project leader, backend developers, and build mechanic developer. On the right side, there is a section titled 'BUILDRA PRIVACY POLICY' which contains the text of the privacy policy. A scrollbar is visible on the right side of the privacy policy text, indicating that the entire policy can be seen.

CREDITS

SHAHER HESS - PROJECT LEADER, BACKEND DEVELOPER

DEREK SHU - BACKEND DEVELOPER

EMILY OU - BACKEND DEVELOPER

JIALU GU - BUILD MECHANIC DEVELOPER

MANGKORN YUAN - AR DEVELOPER, UI DESIGNER & DEVELOPER

BACK

BUILDRA PRIVACY POLICY

LAST MODIFIED 11/26/18

BUILDRA IS A FREE SOFTWARE APPLICATION FOR MOBILE DEVICES THAT WAS CREATED BY JIALU GU, SHAHER HESS, EMILY OU, DEREK SHU, AND MANGKORN YUAN. THIS PRIVATE POLICY STATEMENT EXPLAINS THE PERSONAL DATA BUILDRA PROCESSES, HOW BUILDRA PROCESSES IT, AND FOR WHAT PURPOSE.

WHAT WE COLLECT

INFORMATION YOU PROVIDE TO US

WE REQUIRE CERTAIN INFORMATION TO PROVIDE OUR SERVICES TO YOU. THIS INCLUDES:

Testing Summary: The manual testing for User Story 10 was successful. The spell check website was useful and helped with correct some spelling errors in the Privacy Policy text. The grammar check had some grammar suggestions in the text, however, they were not necessary to change. The user can see their Privacy Policy under Credits in the app and can scroll down to see all of it.

User Story 12 Test:

User Story: “As an administrator, I would like to be able to delete other user creations in the application (if inappropriate)”

Test: Users’ creation is deleted and updated

Testing Procedure:

1. Navigate to the Models page on the admin website
2. Enter a Model ID (2) into the “Enter Model ID” textbox
3. Push Delete
4. Push Refresh Models
5. See if the results are successful and as expected - Model ID 2 should have been deleted from the database and show as deleted on the All Models table

Results:

1. After logging in as an administrator, navigating to the Models page was successful.

The screenshot shows the BUILDRA application interface. At the top, there's a navigation bar with 'Home', 'Users', 'Reports', 'Changelog', and 'Models' (selected). A 'Log Out' button is also present. Below the navigation bar, the page title 'Models' is displayed. The main content area features a section titled 'All Models' with a search bar and a table of models. The table has columns: Model ID, Build Data, Build Name, Username, Location, Privacy, Avatar, and Upvotes. Below the table, there's a 'Refresh Models' button. At the bottom, there's a 'Delete Model' section with an 'Enter Model ID' input field and a 'Delete' button.

Model ID	Build Data	Build Name	Username	Location	Privacy	Avatar	Upvotes
2	{!shipName!:default...	build0	samplename2	samplelocation2	0	0	0
3	{!shipName!:default...	build3	samplename3	samplelocation3	1	0	0
4	{!shipName!:default...	***** replace*****	samplename3	samplelocation3	0	1	0
5	{!shipName!:default...	***** replace*****	samplename3	samplelocation3	0	1	0
6	{!shipName!:default...	aaa	samplename3	samplelocation3	0	1	0
7	{!shipName!:default...	***2	samplename3	samplelocation3	0	1	0
8	{!shipName!:default...	***pack	samplename3	samplelocation3	0	0	0
9	{!shipName!:default...	***pack	samplename3	samplelocation3	0	0	0

Showing 1 to 8 of 8 entries

Previous 1 Next

Refresh Models

Delete Model

Enter Model ID

Delete

2. Entering the Build Name "build0" in the "Enter Model Name" was successful as text could be entered without issues.

The screenshot shows the BUILDRA web application interface. The top navigation bar includes links for Home, Users, Reports, Changelog, and Models, along with a Log Out button. The main heading is "Models". Below this, the "All Models" section displays a table of models. A red box highlights the first row of the table, which has Model ID 2. A red arrow points from the text "Model ID: 2" to this row. Below the table, there is a "Delete Model" dialog box with the text "Enter Model ID" and a text input field containing the number "2". A green arrow points from the text "Delete Model ID Text Box" to this input field. The dialog box also has a "Delete" button.

Model ID	Build Data	Build Name	Username	Location	Privacy	Avatar	Upvotes
2	@shipName!\$default...	build0	samplename2	samplelocation2	0	0	0
3	@shipName!\$default...	build3	samplename3	samplelocation3	1	0	0
4	@shipName!\$default...	***** replace*****	samplename3	samplelocation3	0	1	0
5	@shipName!\$default...	***** replace*****	samplename3	samplelocation3	0	1	0
6	@shipName!\$default...	aaa	samplename3	samplelocation3	0	1	0
7	@shipName!\$default...	****2	samplename3	samplelocation3	0	1	0
8	@shipName!\$default...	****pack	samplename3	samplelocation3	0	0	0
9	@shipName!\$default...	****pack	samplename3	samplelocation3	0	0	0

Showing 1 to 8 of 8 entries

Refresh Models

Previous 1 Next

Delete Model
Enter Model ID
2
Delete

Delete Model ID Text Box

- Pressing "Delete" then "Refresh Models" buttons work as the model is successfully removed from the table.



All Models

Show entries

Model ID	Build Data	Build Name	Username	Location	Privacy	Avatar	Upvotes
2	{!shipName!:default...	build0	sampleName2	sampleLocation2	0	0	0
3	{!shipName!:default...	build3	sampleName3	sampleLocation3	1	0	0
4	{!shipName!:default...	***** replace*****	sampleName3	sampleLocation3	0	1	0
5	{!shipName!:default...	***** replace*****	sampleName3	sampleLocation3	0	1	0
6	{!shipName!:default...	aaa	sampleName3	sampleLocation3	0	1	0
7	{!shipName!:default...	****2	sampleName3	sampleLocation3	0	1	0
8	{!shipName!:default...	***pack	sampleName3	sampleLocation3	0	0	0
9	{!shipName!:default...	***pack	sampleName3	sampleLocation3	0	0	0

Showing 1 to 8 of 8 entries

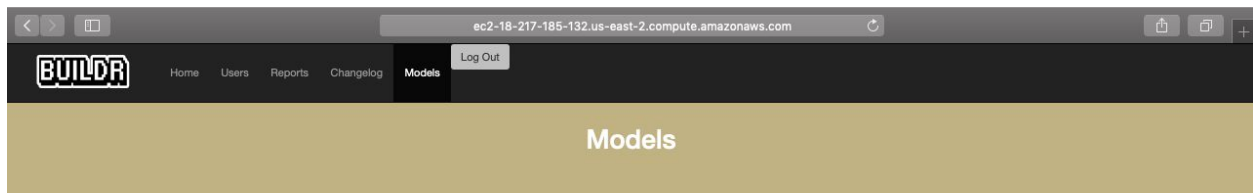
Previous Next

← Refresh Models Button

Delete Model

Enter Model ID

 ← Delete Button



All Models

Show entries

Model ID	Build Data	Build Name	Username	Location	Privacy	Avatar	Upvotes
3	{!shipName!:default...	build3	sampleName3	sampleLocation3	1	0	0
4	{!shipName!:default...	***** replace*****	sampleName3	sampleLocation3	0	1	0
5	{!shipName!:default...	***** replace*****	sampleName3	sampleLocation3	0	1	0
6	{!shipName!:default...	aaa	sampleName3	sampleLocation3	0	1	0
7	{!shipName!:default...	****2	sampleName3	sampleLocation3	0	1	0
8	{!shipName!:default...	***pack	sampleName3	sampleLocation3	0	0	0
9	{!shipName!:default...	***pack	sampleName3	sampleLocation3	0	0	0

Showing 1 to 7 of 7 entries

Previous Next

Delete Model

Enter Model ID

Testing Summary: The manual testing for User Story 12 was successful. The administrator can login and navigate to the models page without any issues. The Model ID "2" was typed into the "Enter Model ID" textbox and after pressing the delete then refresh models buttons, the model was deleted from the table.