Form Buster
Team Members:

- Daniel Acosta (dacosta2022@my.fit.edu)
- Christopher Demuro (cdemuro2022@my.fit.edu)
- Alex Merino (amerino2022@my.fit.edu)
- Luka Miodrag Starcevic (lstarcevic2022@my.fit.edu)

Faculty Advisor: Phillip Bernhard (pbernhar@fit.edu)

Client: Phillip Bernhard (pbernhar@fit.edu)

Progress of Current Milestone:

Task	Completion	Daniel	Chris	Alex	Luka	To-Do
Implement, Test Form Tracker	95%	0%	0%	100%	0%	GUI cleanup
Implement, Test Active Forms	80%	0%	0%	100%	0%	Figure out logic for who needs to sign a form
Implement, Test Signature Graph	60%	0%	0%	100%	0%	Make graph responsive, show details when hovering a signature
Implement, Test Form Autofill	50%	0%	0%	100%	0%	Standardize form input names across all form types
Implement, Test User Authentication	100%	0%	100%	0%	0%	None
Implement, Test, Demo Admin Deletion of Form	100%	0%	0%	100%	0%	None
Implement, Test Inbox Messages	60%	100%	0%	0%	0%	Add reading of messages, clearing of inbox, refresh button, send notifications based on signee graph

Discussion of Accomplished Tasks:

- **Implement, Demo Form Tracker:** During this milestone, the Form Tracker implementation began with the backend api. We added three new api functions that grab data from the database to display in the Form Tracker and its subcomponents. We removed the hardcoded values from last semester and instead used the Form component to fill out a form and display in the Form Tracker as it would in production. Another task included overhauling the database schemas to account for an earlier implementation bug which caused forms to duplicate in the tracker as well as storing objects in other database objects, instead of just ID references, which is now the current implementation.

- **Implement, Test Active Forms:** During this milestone, every function needed for a user to fill out a form and submit it to the database correctly was finished. The logic behind which users need to be attached to a certain form is still being implemented. Each user that is attached to the form is given their own signature object with a unique ID, which is connected to the form object, and has the form ID attached to their account. This allows for an easy security verification when any user attempts viewing specific form details so we do not violate any privacy laws. With all functions in the backend properly working, the correct forms populate a user's account when a new form is created.
- **Implement, Test Signature Graph:** The signature graph began implementation this milestone with the goal of getting a proof of concept demo complete before the milestone ended. This goal was completed and for each form in the database, a graph of all signatures attached will show up in a representation of a tree-like data structure. This allows for users to see the parallelism of signatures for their form as well as which users have or have not signed the form.
- **Implement, Test Form Autofill:** Autofill of form data based off user information has been a main feature of the entire system since conception of the idea and this milestone is when it was finally put into action. When a user opens a form, in predefined fields such as *name, school_id,* and *date* we pull the data from the user's account (given to us from their registration) and fill in the respective fields. This allows for users to complete forms quicker than the current PDF system. The only field that will not be autofilled is the *Signature* field since users must sign the form themselves.
- **Implement, Test User Authentication:** Prior to this milestone, our application did not properly handle user authentication and sign-in. It was unable to differentiate between users, and would not prevent a user from signing in when they provided an incorrect password. Both of these issues were resolved during this milestone, and proper user authentication has been implemented. It is still the recommendation of the team that the included login system should be replaced by the centralized login system used by the target university, but for demonstration purposes, the current included system is more than adequate.
- **Implement, Test, Demo Admin Deletion of Form:** During this milestone we set aside some focus on adding a feature specific to administrators. Since we are constantly adding new active forms to the database in testing, we decided that to make it easier to delete active forms with the GUI. Administrator users can see every active form in the database the same way any user can see the forms that are attached to their account. The administrators are given an extra button which allows them to delete a form. When the button is clicked the form is removed from the database, followed by the signatures attached to the form, followed by the form from each user's account. This removes all traces of the active form from the database with no way of recovering them.
- **Implement, Test Inbox Messages:** During this milestone, the database was expanded to allow for users to receive notifications when a form has been filled out and when it has

been signed. When a user creates a new form, all signees will be notified and receive an inbox message that can be seen when viewing the inbox. When a form is signed and completed (or rejected), the form creator will be notified of the result.

Discussion of Contribution of each Team Member:

- **Daniel Acosta:** Tasks contributed to this milestone include implementing the Inbox notification system. The backend functions for the inbox notification system were implemented, tested, and connected to the frontend GUI. A new set of schemas in the database were configured to accommodate notifications (including the message ID, if the message was read, if it was rejected, etc.). Users are tied to a list of notification message IDs and the inbox is populated with all notification messages tied to that user's account.
- **Christopher Demuro:** Contributed to this milestone by implementing and testing user authentication, which included bug fixes in the password checking and authentication code, as well as properly implementing the per-user sign in process. These were required both to keep user accounts secure, and to ensure that the application could differentiate between individual users and their active forms, as well as what notifications should be delivered to each user.
- **Alex Merino:** Tasks contributed to this milestone include implementing the Form Tracker, Active Forms, Signature Graph, Autofill, and Admin Form Deletion. The backend functions for the Form Tracker were implemented, tested and connected to the frontend GUI along with the Active Forms that populate the Form Tracker. These components were worked on hand in hand for them to be able to function in cohesion. Another task was beginning the implementation of the signature graph as well as creating a proof of concept demonstration of how a potential signature graph could look like in the GUI. Autofill was another task that was implemented where comparing the user account information and form inputs was needed to add cases for where inputs should be autofilled. A simple check of the form inputs is done and if there is a match the user account information is filled. The last task completed was the Admin Form Deletion, where new backend functions were implemented to delete active form data from the database. These backend functions were then connected to the frontend, but specifically for administrators only, where they can look over all forms and delete ones that they choose.
- **Luka Miodrag Starcevic:**

Milestone 5 Task Matrix:

Task	**Daniel**	**Chris**	**Alex**	**Luka**
Implement, Test, Demo Form Editor	0%	0%	0%	100%
Implement, Demo, Test Updating a Form	0%	100%	0%	0%
Implement, Demo, Test Forms in the Inbox	100%	0%	0%	0%
Implement, Test Signature Graph	0%	0%	100%	0%
Implement, Test Completing a Signature	0%	50%	50%	0%
Conduct Evaluation and analyze results	25%	25%	25%	25%
Create Poster for Design Showcase	25%	25%	25%	25%

Discussion of Planned Tasks:

- **Implement, Test, Demo Form Editor:** Continuing the completion of fully-functional form editor, including support for creating and managing form templates, adding content, including, but not limited to, text, formatting and other content, and saving and retrieving form templates for modification and use. Add form preview to editor so that changes to forms can be seen in real time as they are being made. Tests will be performed to ensure that all form templates are saved and recalled reliably.
- **Implement, Demo, Test Updating a Form:** Implement the updating of an already existing template. Currently, existing templates can be opened in the form editor, but cannot be modified correctly. Further testing and features are required to ensure template modification is working properly.
- **Implement, Demo, Test Forms in the Inbox:** Implement the retrieval of forms sent to the inbox after completion. This includes form approvals/disapprovals for students/faculty, as well as the notification of new forms that need to be signed in faculty inboxes. Form approvals/disapprovals in the inbox should have an optional attached note explaining the form's outcome. Tests will be performed to ensure that the correct users only see the forms meant for their user level (i.e. no students should receive submitted forms, only faculty).
- **Implement, Test Signature Graph:** Continuing from the progression made from last semester, the Signature Graph will be implemented to be web responsive as well as be able to handle parallelism in form signatures. Another feature that will be added is where

when a user hovers over a node, the information of that node will be displayed on the side of the web page. Testing will be complete to ensure the completeness, accuracy, and dependability of the feature.

- **Implement, Test Completing a Signature:** A new page will be implemented on the site where a user who has not yet signed a form can sign or decline a form. This page will update the form along with the user's signature object in the database. The page, with specific form data, will only be available to those that are connected to the form. This page will be tested for security, accuracy, and reliability to ensure that no signature is forged, incorrect, or lost in the database.
- **Conduct Evaluation and analyze results:** We will begin our evaluations of the system via giving people who have never seen the website a list of tasks to complete for each user type and evaluate them on speed and amount of help they need. We will then ask for feedback on how easy or difficult each task was via a google form and analyze those results to see how we can improve our accessibility.
- **Create Poster for Design Showcase:** We will begin creating the Senior Design Showcase Poster for the project, where we will add the necessary information needed and organize the poster in a neat and coherent way.

Dates of meeting:

- 9/29/2025

Client feedback on current Milestone:
See Faculty Advisor Feedback below.

Faculty Advisor Feedback:

- Implement, Test Form Tracker:
- Implement, Test Active Forms:
- Implement, Test Signature Graph:
- Implement, Demo, Test Adding comments to Forms:
- Implement, Test User Authentication:
- Implement, Test, Demo Admin Deletion of Form:
- Implement, Test Inbox Messages:

Signature: ________________________________ Date: ________

Evaluation by Faculty Advisor:

- Faculty Advisor: detach and return this page to Dr. Chan (HC 209) or email the scores to pkc@cs.fit.edu
- Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

Daniel Acosta	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Christopher Demuro	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Alex Merino	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Luka Miodrag Starcevic	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

- Faculty Advisor Signature: ________________________________ Date: ___________