Milestone 2 Progress Evaluation

Team Members

Grace Dolphy - gdolphy2015@my.fit.edu
Raphael Setin - rsetin2015@my.fit.edu
McNels Sylvestre - msylvestreph2016@my.fit.edu

Faculty Sponsor

Dr. Eraldo Ribeiro - eribeiro@cs.fit.edu

Client

Dr. Philip Chan - Computer Engineering and Sciences at FIT - pkc@cs.fit.edu

Meeting(s) with Faculty Sponsor

10/25/18 (3:30 PM - 4:20 PM)

Meeting(s) with Client

10/12/18 (5:00 PM - 5:25 PM), 10/26/18 (4:00 PM - 4:53 PM)

Progress of Current Milestone

Task	**Completion %**	**Raphael**	**Grace**	**McNels**	**To do**
1.Implement pseudo-TRACKS system	100%	33.3%	33.3%	33.3%	None
2. Implement pseudo-PAWS database	100%	33.3%	33.3%	33.3%	None
3. Implement Form Buster database	100%	33.3%	33.3%	33.3%	None
4. Implement home page for the different categories of users	100%	33.3%	33.3%	33.3%	None
5. Implement student records page	100%	33.3%	33.3%	33.3%	None
6. Implement forms catalog page	100%	33.3%	33.3%	33.3%	None

7. Implement form submission pages	100%	33.3%	33.3%	33.3%	None

Discussion of each accomplished task (and obstacles) for the current Milestone

- **Task 1:** Implementing a pseudo-TRACKS to log in users to Form Buster was difficult, as each member attempted it. When doing further research, we realized that Firebase does not support authentication over different domains. Therefore, we created a subdomain under our “formbuster.me” domain, named “auth.formbuster.me”. Upon authentication, a user is redirected from the subdomain to the top domain. Another issue was making it possible for users to authenticate with just a TRACKS ID, as opposed to typing the full email as their username. A fix has been found for this, and will hopefully be ready by the next milestone.

- **Task 2:** It was important to figure out what type of information to save for each user in the database in order to come up with a realistic simulation of the actual PAWS database. Therefore, more time was spent on researching first what is currently saved in PAWS, and also what would be needed from a university database for the purposes of the project. Since we are using a database that follows a NoSQL scheme, we decided to have a “Users” collection as our top level node, as opposed to have multiple collections that’d correspond to each category of users (Students, Faculty, Staff and Student Coordinators). Then, under the “Users” collection, we have documents, each of which identified using a user’s TRACKS ID. The rule adopted is a) an ID that contains a year is a student ID (ie: aadkins2016), b) if not, then it’s either one of the remaining types of users. The actual type can be verified by accessing the *userType* field within the document. Inside of a document (each representing a user), personal information such as name, address, major, email, advisor and userID can be found. The pseudo-PAWS database also currently holds deadlines that will automatically be added to a form upon submission (those may be overridden later on). This might be temporary or permanent depending on whether it is possible to fetch those deadlines from an API endpoint.

- **Task 3:** Form Buster’s own database contains two main collections: one for form templates and one for completed forms for users. The *formTemplates* collection is where the fields of each different form live. It’s also where the different versions of the same form can be found. The ID of a form template is such that the title and the creation date of the template is readily available (ie: *registration_10282018225900*). All the fields there are empty, and queried by the webpages upon request to fill out a specific form. For each template, placeholders are placed for approvals (who signed, when he/she signed, and why he/she did not sign, if applicable). At this point of the development cycle, the focus is on the registration form, so fields to take in courses information are present as well (crn, title, credits, etc). The *users* collection contains documents that represent each a specific user, within which is found a collection of the forms they have submitted (organized under *inProgressForms*, *drafts* and *completedForms*).

- **Task 4:** All user roles have appropriate pages displayed, as well as appropriate menu items. An obstacle was figuring out how to apply the same HTML code to many areas without having to copy and paste the same code into many files, JQuery's load function was sufficient, as the data on these pages did not need to be changed and didn't require a for loop, and we could use CSS to hide what we did not want to show on a role's page. In order to populate the dashboard's in-progress forms for a student, the only solution we found was to write up a JavaScript function, that would produce the equivalent of our HTML code, but it was not easy to construct and it's something that would be hard to maintain. With more research, we later realized we could use jQuery's append function and pass the HTML in as a string, to save time; this has been applied to the notifications, and at a later time, we will apply this technique to the in-progress forms.

- **Task 5:** The page that allows a staff/faculty/student coordinator to view the list of all students is ready, we will populate this page with our database data in our next milestone. We will also implement in the next milestone the backend part for searching for students in the Form Buster's database, and for accessing their form history as well.

- **Task 6:** The page that allows a student to view the forms available and select a form to start is completed. A search bar at the top of the page allows for input of search keywords to filter the results, which are queried from the Courses API on the backend. At this stage, only filtering by title will be implemented but future plans will include searching by a string from the description or a combination of parameters.

- **Task 7:** The form submission page implemented only focuses on a student starting a registration form (by picking it from the list). The user will then be brought to the 1st page of the form. This will lay out our foundation for a later milestone, when we will allow the student to actually search for and select classes to add to the form, and continue on to the remaining portion of the form and its submission.

Discussion of contribution of each team member to the current Milestone

- **Grace:** Worked on the front end of each user role using CSS, JavaScript, HTML and JQuery. Generalized code for reuse as well as to prepare for the inserting of variable information from the database onto a page. Researched Firebase authentication.

- **Raphael:** Mainly worked in the backend, especially with the Firebase database. Designed the structure of FormBuster, pseudoPAWS, and pseudoTRACKS database with McNels. Researched Firebase documentation in order to have a better understanding on how to write and read data from the collections and documents of the Firestore. Helped with the initialization of the databases in the project and implemented a JavaScript file to randomly populate fake users in all three databases, with the help of some fake data obtained from the internet. Lastly, contributed in the in-progress forms page of students by implementing functions for populating the HTML fields with the user's data from Form Buster database.

- **McNels:** Set up Firebase projects for authentication, university database and Form Buster database. Investigated importing users in bulk and using custom ids for authentication. Investigated implementation of a custom authentication system with CAS using Ruby. Investigated W3.CSS framework to improve the webpages' look and feel. Implemented authentication system and sign out function on a single domain (hopefully subdomain soon). Contributed to the organization of the source files for the website pages, and the structure of the methods and functions to be implemented. Contributed to the design and structure of the databases, with regards to storage of users and forms. Integrated the separate components of the system into one.

Plan for the next Milestone

Task	**Raphael**	**Grace**	**McNels**
● Implement, test, and demo digital approval for Students and Faculty	33.3%	33.3%	33.3%
● Implement, test, and demo database lookup for student records	33.3%	33.3%	33.3%
● Implement, test, and demo email notifications feature	33.3%	33.3%	33.3%
● Implement, test, and demo form tracking feature	33.3%	33.3%	33.3%

Discussion of each planned task for the next Milestone

- <u>Implement, test, and demo digital approval for Students and Faculty:</u>
  To achieve this one, we will need to complete the implementation of the “Start a form feature” on the students side. The form we will be using is the Registration Form, and we will be using a pre-filled form where we populate the information into our database. On the faculty side, the form will need to be populated into the appropriate faculty member’s dashboard, where they can view the form, and select to approve. The form will then be removed from the faculty members dashboard.

- <u>Implement, test, and demo database lookup for student records:</u>
  To achieve this one, we will need to first populate the student’s list that can be accessed by Staff, Faculty, and Student Coordinator. Then once a student is selected, we will allow the previous mentioned users to see a list of history forms on the site, which for now will be manually added into the database.

- <u>Implement, test, and demo email notifications feature:</u>
  To achieve this one, we will send notifications based on the tasks in Task 1. The first will be notifying the faculty member they need to sign a form, and the second one, when the faculty has signed the form, the student should be notified. In order to test this feature, all three of us will have to use our own emails in order to test if the email was successfully received or not, since the emails that are currently in our database are not legitimate, thus preventing us from using them.

- Implement, test, and demo form tracking feature:
  To achieve this one, we will again use the action in Task 1. Once the faculty member has approved the form, the student’s dashboard should update with a green check mark.

Sponsor feedback on each task for the current Milestone

- Implement pseudo-TRACKS system:
  Time out the user after 30 minutes of inactivity, too much time is not an issue, but timing out too soon may annoy the user. If a user closes the tab or browser with FormBuster logged in, sign out the user.

- Implement pseudo-PAWS database:
  The pseudo-PAWS schema was great; nothing to change.

- Implement Form Buster database:
  The form Buster database was also great. The only thing that should be considered is whether or not is a good idea to maintain the email field for each user, as that can be easily obtained from pseudo-PAWS database.

- Implement home page for the different categories of users:
  Looks good at the moment, but keep in mind features are important.

- Implement student records page:
  Looks good at the moment, but keep in mind features are important.

- Implement forms page:
  Looks good at the moment, but keep in mind features are important.

- Implement form submission pages:
  Looks good at the moment, but keep in mind features are important.

Sponsor Signature: ________________________________ Date: ________

Sponsor Evaluation

Sponsor: detach and return this page to Dr. Chan (HC 322)

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)

Grace	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Raphael	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
McNels	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

Sponsor Signature: ________________________________ Date: ________