**CEN 4010 - Principles of Software Engineering**

**Summer 2021 - Full Term**

**Milestone 1 Project Proposal and High-level description**

Team 8

*Burrow*

Team Members

Christian Engman - *cengman2017@fau.edu*

Benjamin Thaw - *bthaw2018@fau.edu*

Raymond Budoff-Kingsland - *rbudoffkings2018@fau.edu*

Nikita Ostro - *nostro2018@fau.edu*

Garrett Reardon - *greardon2018@fau.edu*

Documentation created on **June 15, 2021**

History Table

|  |  |  |
| --- | --- | --- |
| **Version #** | **Revision Date** | **Changes included** |
| 1 | 6/15/2021 | Initial version |

# Table of Contents

**Section Title** **Page Number**

[**Table of Contents**](#_x7p22amgxmtz) **2**

[**Executive Summary**](#_j4tc6pkwpbpj) **3**

[**Competitive Analysis**](#_xz0ljxcrdafe) **3**

[**Data Definition**](#_j8r75ytro3ok) **4**

[**Overview, Scenarios, and Use Cases**](#_ok903y2ma4lz) **6**

[**Functional Requirements**](#_q0e4u17666sz) **7**

[**Non-Functional Requirements**](#_fvr10j6gcox) **12**

[**High-Level System Architecture**](#_sh0j55wi4889) **14**

[**Team**](#_y0mpwlnvnxzt) **15**

[**Checklist**](#_7kbozw94a26x) **15**

# Executive Summary

While completing course work at universities, students thrive through group work and collaboration. In the past year, the opportunity to work collaboratively with other students has diminished due to the COVID-19 pandemic. Many students have turned to social media services to communicate with fellow students, while some have continued through their course work without the normal peer collaboration and communication that normally occurs.

The proposed application, *Burrow*, will be developed to help close the gap in student peer-to-peer collaboration during online instruction. While many social media platforms exist currently, none are primarily geared towards supporting university students, and are instead very generic or broad.

# Competitive Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Burrow** | **Discord** | **WhatsApp** | **iMessage** |
| **Send Messages** | ✓ | ✓ | ✓ | ✓ |
| **Channels for courses/groups** | ✓ | ✓ | X | ✓ |
| **Advanced moderation capabilities for instructors** | ✓ | X | X | X |
| **University and Instructor Announcements and Alerts** | ✓ | X | X | X |
| **Designed specifically for academic use** | ✓ | X | X | X |

The advantage that Burrow will offer over its competitors for its advertised use case is its tailored design towards the academic environment. As courses have moved online, text communication platforms like WhatsApp and Discord have proved very useful to students for cultivating a social community around classes and allowing students to discuss coursework. Unfortunately, The fact that students use independent third-party platforms for this purpose has proved problematic.

If instructors are unaware of these chats, they will have no ability to regulate the behavior of their students in the ways they could inside of a normal classroom. This, among other factors, has led to a spike in cheating and other forms of academic dishonesty as virtual classes have been made the norm over the past year.

Burrow seeks to combat this by providing a safe, easy to use environment with an experience similar to these other messaging applications, while allowing instructors and administrators to regulate what happens in these chats and stop cheating in its tracks.

Burrow will also be able to send announcements and alerts from instructors and administrators, allowing students to more easily see information they might pass over in their inbox.

We hope that Burrow will be a compelling choice for both instructors and students alike, allowing them to connect with their peers in a safe, secure, and convenient environment, while still upholding the values of Florida Atlantic University and other institutions that wish to use the service in a way that our competitors simply cannot because of their desire to have as many general uses as possible.

# Data Definition

|  |  |  |
| --- | --- | --- |
| **Term** | **Category** | **Definition** |
| University | Entity - Organization/University | The container for all channels, messages, and users belonging to a specific university, such as FAU |
| University Staff  University Administrator | Entity - Person | Users in a University who have permission in this application to manage settings, Users, Channels, and content |
| Student | Entity - Person | Users that are enrolled in courses at a University |
| Instructor | Entity - Person | User account for professors/instructors of a course |
| User  Account | Entity - Person | An individual who uses the system and has logged in |
| New User | Entity - Person | An individual who has not yet logged into the system or created an account. |
| Unverified User | Entity - Person | A User who has not signed into Google to verify they own the email address |
| Channel | Collection - Data | An area where Users can send Messages and where Instructors can send Announcements |
| Message | Entity - Data | A text string of user-generated content, as well as related metadata including authors and dates sent |
| Message Author | Entity - Data Attribute | The User who created a message |
| Group  *Group Channel* | Collection - Data  Type: Channel | A type of Channel with members assigned by Instructors, designed for group-based projects or assignments |
| Report | Entity - Data | A way to flag Messages or alert Instructors or University Staff about issues. |
| Global | Description | Something that applies to all Channels and all Users |
| Course Channel | Description  Type: Channel | The generic type of Channel designed for a specific course section |
| Announcement | Entity - Data  Type: Message | A certain type of Message that is highlighted for Students |

# Overview, Scenarios, and Use Cases

*Burrow* will have many features of traditional messaging platforms, including the ability to send messages, organize communication channels, and support group discussions, but also will be able involve instructors and university staff directly in the application for creating channels for course sections, creating group discussions for students working in groups, and the ability to moderate content to detect and prevent spam and academic dishonesty.

*Burrow* will organize conversations into channels for specific sections. Students will initially be able to join channels by selecting the courses and sections they are registered for during the current semester, but future support may include automatically importing such data. Instructors will have the option to add or remove students manually from courses they teach, as well as to create separate groups so that students who are working on group projects can communicate with each other.

*Burrow* will primarily be targeted towards students at Florida Atlantic University, but can be expanded to other universities, schools, and communities if there is interest. By starting with FAU, we can implement features that can have a greater impact on our own communities.

Scenario & Use Case Stories

* **Registration**: A student registers on the website, verifies their email address by signing into Google, and selects the courses they are currently enrolled in.
* **Exploring Channels**: A logged-in user will be able to see a list of channels they have access to see and send messages into in the user interface. The list of channels is divided into Course Channels and Groups. Clicking on a specific Channel or Group will display the message on the web page. They can scroll up to load older messages. Clicking on another channel replaces the old channel’s messages with those from the clicked on channel. New messages are loaded every 5 seconds, or when a message is sent.
* **Sending Messages**: A logged-in student can type messages into the message box after clicking on a channel. After clicking the Enter key or the Send button, the message will be sent into the channel. A logged-in instructor or staff member will see a checkbox next to the send button, which will allow them to mark their message as an announcement.
* **Editing and Deleting Messages**: A logged-in user will be able to click the “More” button (represented by an appropriate icon) on their message, giving them the option to Edit and Delete the message. Clicking on the edit button will show a pop-up to let them change the contents of their message, along with a warning that the old version will be preserved. Clicking the delete button will show a pop-up asking for confirmation and showing a warning message. Instructors and staff will be able to mark or unmark a message (even if not their own) as an announcement.
* **Viewing Announcements**: Announcements will appear in the normal message stream as highlighted, but a logged-in user can click a button at the top of the UI to view a list of previous announcements.
* **Reporting Messages**: Clicking the “More” button on any message will include an option to Report the message. A popup will appear, asking the user to select a report reason and enter an option description. The user will click the “Submit” button to submit the report.
* **Viewing Reports**: A tab will appear for moderators (instructors and university staff) to review reports. The moderator will be able to comment privately on the report, Ignore the report, delete the message reported, send a warning message to the user, mute the user temporarily, or permanently ban the user from the course or the entire service.

# Functional Requirements

Functional requirements below will be in an ordered list. These requirements can be referenced as REQ-F# (eg: REQ-F3 or REQ-F2.3).

1. **Account Creation**

Most system functions will require the User to create or log into an account. New Users can create an account by entering their FAU email address, First Name, Last Name, and a password. New Users will also be able to choose from a set of predefined options for displaying their name.

A random, unique identifier will be generated by the server, which will be used to identify Users in request responses without the need to expose the User’s email address.

If an account with the email address already exists, they will be prompted to sign in instead.

The New User will need to agree to any terms and privacy policies provided.

1. **Google Account Sign in**

Due to the school-based nature of this application, Unverified Users will be asked to sign into a Google account after registering for an account.

The Unverified User will be asked to sign into Google to verify that they own the email address they signed up with. The user ID of the Google account will be stored along with the User, which will allow the user to reset the password in the future if needed.

Users who attempt to sign in (REQ-F3) but have not signed in with Google will be asked to sign in again.

1. **Account Sign in**

New Users who have accounts but are not signed in will need to sign in to access most functions of this application. The sign in process will involve the user entering their email address and password.

If the email address and password combination match, the user will be signed in. Otherwise, an error message will be displayed.

If the user enables Two Factor Authenticator (REQ-F13), the user will be prompted to enter a 6-digit code generated by their authenticator app before the sign in is completed.

1. **Customizable Profile/Bio**

The user will be given a limited set of options to customize their profile, which will be shown to other users.

Some options may include how to display their name (based on their first and last name), whether to list the courses they are taking, and whether they want to select and choose their major(s) and any minors.

Depending on storage space availability or the implementation of another cloud storage provider, profile pictures may be allowed at a later point in the lifespan of this application.

1. **Announcements**
   1. **University Announcement**

University Administrators and Staff will have the option to send Announcements out to all logged-in Users. These announcements will appear in their own Channel at the top of the User’s Channel list.

* 1. **Course Announcement**

Instructors can write Announcements in Course Channels so that all students in their section can see it. Announcements will be typed as a normal Message, but the Instructor will have an option to make the Message an Announcement before or after sending it. These Announcements will appear in the chat in a highlighted color and will also be accessible in an Announcement tab for the Channel for future access.

1. **Moderation Capabilities**

This system will have advanced moderation capabilities geared at allowing Instructors to prevent cheating and creating a safe, online community.

* 1. **Message Reporting**

Users will be able to report other Users’ Messages for reasons including academic dishonesty, inappropriate netiquette, and general misconduct.

Moderators (University or Instructors, see REQ-F6.3) can choose to respond to a report by banning a User, deleting a Message, and/or sending the user a warning. Moderators can comment on reports they have access to.

* 1. **Automatic Message Flagging**

The system will attempt to flag (by reporting, REQ-F6.1) messages that contain specific keywords that may indicate cheating, academic dishonesty, or potentially harmful material.

* 1. **Moderation Actions**

Instructors and University Staff have certain moderation actions available to them. Instructors have access to moderation features within their own Course Channel, for messages reported in their course, or for Groups and Group Messages that belong to their courses. University staff have access to moderation features Global (for any course or group).

Moderation actions include removing students from courses (also removes them from Groups in REQ-F13), deleting Messages (REQ-F11), and temporarily muting (REQ-F6.4).

* 1. **User Muting**

A muted user is unable to speak in the Channel they are muted in.

Mutes are temporary (expire at a specified time), and can be enforced in a specific Channel or Globally (in every Channel).

Instructors can mute students in their own course Channels, while University Staff can mute students in Channels or Globally. During the mute, students will be unable to send (REQ-F9) or edit (REQ-F10) Messages but can still read Messages (REQ-F8).

1. **Course Selection**

Upon registration or at the start of a new school semester, students will be asked to select courses that they are taking.

Instructors will be able to manually add students, TAs, or tutors to their course if they are indicated as the owner of the course section.

1. **Read Messages**

Users in Channels are able to read Messages in Channels they are a part of, including Course Channels and Group Channels.

Users will not be able to read Messages in Channels they do not have access to.

1. **Send Messages**

Users in Channels are able to send Messages in Channels they are a part of, including Course Channels and Group Channels.

Users will not be able to send Messages in Channels they do not have access to, and cannot send Messages if they have been muted (REQ-F6.2).

1. **Edit Messages**

Users are able to edit their own Messages. The Message prior to the edit will be logged to prevent Users from hiding evidence of misconduct.

Users cannot edit Messages if they have been muted.

1. **Delete Messages**

Users are able to delete their own Messages, and Instructors and University Staff are also able to delete Messages. Deleted Messages are logged to prevent users from hiding evidence of misconduct.

Instructors can delete Messages in their own Course Channels or Group Channels sent by any User. University staff can delete Messages sent in any Channel by any User.

1. **Groups**

Groups are smaller Channels that contain a limited number of students. These Group Channels are a separate type of Channel that only assigned members and Instructors can access and interact with.

* 1. **Group Creation**

Instructors can create empty Groups for each desired group in their course. Instructors can delete Groups at any time.

* 1. **Groups - User assignment**

Instructors can assign users to certain Groups. Instructors can remove users from Groups at any time.

1. **Two-factor Authentication (TFA)** *(low priority)*

Two factor authentication may be implemented to protect Accounts. If implemented, Users will have the choice to enable this feature.

Two factor authentication adds an extra layer of security by requiring Users to enter a 6-digit code generated by a supported authenticator app (such as Google Authenticator) after entering their email address and password to verify that they are the user they’re claiming to be.

If implemented, Users will be able to modify their TFA settings and scan a QR code on their Authenticator app. They will need to provide the 6-digit code before TFA can be enabled, and every future sign-on will require it.

# Non-Functional Requirements

Most non-functional requirements based on performance, expected load, fault tolerance, availability, and storage availability are highly dependent on the LAMP server. Our team does not have the ability to control these factors (we don’t own or manage the server), but we can optimize some things such as file size and SQL queries.

These functional requirements may change in the future after testing on the LAMP server to determine current performance.

1. **Cross-Browser Compatibility**

This application will run and display properly on the latest version of the 3 web browsers: Google Chrome, Mozilla Firefox, and Safari (macOS).

Issues in this requirement can be identified by non-working functions or UI inconsistencies among browsers.

1. **Cross-Device Compatibility**

This application will be designed to display and function properly on mobile devices in addition to desktop computers and laptops. The application will still need to be accessed through a web browser but should have a design consistent to that of the desktop version.

1. **Data Storage**

All data in this application will be stored on the LAMP server’s MySQL database.

1. **User Privacy**

Every effort shall be made to protect user privacy.

Data access will be restricted to users who have access to the data. Privacy-related issues should be reported immediately to a developer so it can be fixed at a high-priority. These privacy issues must be acknowledged within 48 hours and resolved within 96 hours.

Privacy policies shall be linked at the bottom of each page and displayed upon account creation to communicate to users how their data will be used.

1. **Language and localization**

The primary and only language of web page content (content created by the developers) will be English.

User generated content may include non-English language text.

1. **Security**
   1. **Registration and Login**
      1. All messages will be accessible only to users who have logged in.
      2. New users must sign in with Google to verify that they own the email address they provided.
   2. **Passwords**
      1. Passwords must meet minimum length and complexity requirements, to be defined at a later point.
      2. Users will be warned not to use the FAU passwords.
      3. Users can only reset their passwords by signing into their Google account, if it has already been linked.
   3. **Encryption** (*low priority*)

Encryption helps keep information secure while at-rest and in-transit.

The LAMP server already utilizes HTTPS to encrypt information in-transit.

Encryption may be implemented for data at-rest with a symmetrical public and private key, although it may be difficult to properly test and implement, as well as to store the private key securely (due to the nature of the LAMP server).

1. **Design and User Interface**

The application shall be composed of an adequate and consistent design. Most UI elements will utilize the Bootstrap framework.

1. **Search Engine Optimization (SEO)**

Appropriate HTML tags will be included in web pages to ensure the application is accessible and searchable by major search engines.

To protect user privacy, user generated content (messages, channels, user information) will not be included (because they will require authentication).

1. **Performance**

*These requirements may change after further testing on the LAMP server’s existing services*.

* 1. MySQL queries shall be optimized to take a maximum of 500 ms - 1 second to complete.
  2. Maximum file size shall be 1 MB to reduce load and render time.
  3. API Read and List requests shall take a maximum of 2 seconds.
  4. API Create and Update requests shall take a maximum of 4 seconds.
  5. Page load time (before data is loaded) shall be a maximum of 2 seconds.

# High-Level System Architecture

* MySQL
  + Used for the Database
* HTML/CSS/JS (vanilla)
  + Used for the actual visual interface interface
* PHP
  + Used as backend language/platform
* LAMP Server provided by FAU
  + Used to host the MySQL database and PHP server
* Git & GitHub
  + Used for version control
* Jira Scrum
  + Used to organize team efforts and progress in sprints
* Bootstrap
  + Used to create UI designs and a mobile-friendly/mobile-first design
  + License: (MIT) <https://github.com/twbs/bootstrap/blob/main/LICENSE>
* Google API Client PHP Library
  + May be used to ensure that users who sign up with email/password actually own their email account
  + <https://github.com/googleapis/google-api-php-client>
  + License: (Apache 2.0) <https://github.com/googleapis/google-api-php-client/blob/master/LICENSE>
* GoogleAuthenticator Library for Two Factor Authentication
  + May be used to implement two factor authentication for added security using the Google Authenticator (or similar) mobile apps
  + <https://github.com/sonata-project/GoogleAuthenticator>
* Other libraries will be determined during development and implementation of features

# Team

Raymond Budoff-Kingsland - **Product Owner**

Christian Engman - **Development Team**

Nikita Ostro - **Scrum Master**

Garrett Reardon - **Development Team (Backend Lead)**

Benjamin Thaw - **Development Team (Frontend Lead)**

# Checklist

|  |  |
| --- | --- |
| **Status** | **Item** |
| DONE | Team decided on basic means of communications |
| DONE | Team found a time slot to meet outside of the class |
| DONE | Front and back-end team leads chosen |
| DONE | Github master chosen |
| ON TRACK | Team ready and able to use the chosen back and front-end frameworks |
| DONE | Skills of each team member defined and known to all |
| DONE | Team lead ensured that all team members read the final M1 and agree/understand it before submission |