American Astronautical Society

Field Session Team

Summer 2018

Emma May
Antonia Sisneros
Jake Wong
Jeff Greene

I. Introduction

The American Astronautical Society (AAS) is a national society with chapters across the country dedicated to advancing space research and connecting those across the diverse field. The Rocky Mountain Section hosts a conference every year for aerospace companies in the region to gather and share their research. They have an outdated website, and no way to connect conference attendees. The committee members tasked with planning the conference also have a huge job of reviewing all submitted abstracts to determine which corresponding papers will be presented at the conference. Currently, presenters must email individual committee heads with their abstracts. The heads must then review the individual abstracts and personally keep track of which they accept, and inform the author that they may present the corresponding paper at the conference. The Rocky Mountain section requested this field session team update their website with a few primary goals: add functionality to their website to create user accounts, user login, an interface for publication submissions and review by Committee members, create a database with a CRUD (create, read, update, destroy) wrapper, and modernize the aesthetics of the site.

II. Requirements

Given the project scope and vision from the client, the field session team created the following list of functional and non-functional requirements.

Functional requirements of the site, as agreed upon by the client and the team are as follows. The AAS website shall be a dynamic, database-driven website where the content of the website is directly supplied by the database. Features that had to be incorporated include:

- User account creation
- User login
- Publication submission form for members only
- Publications to be read only by members who registered for the conference at which it was presented
    - Search functionality to filter publications to view
- Publication review by committee members to accept submitted abstracts

Given these functional requirements, the field session team also generated a list of non-functional requirements that must be met for the project to be accepted by the client and considered complete. These requirements include:

- Acceptable performance speed
- Scalable to use all provided content
- Maintainable - server should be able to host an appropriate amount of users
- Coding style - code should be well commented and readable
- Aesthetically pleasing and acceptable style, decided upon by the client
- Documentation of functionality and user’s manual so additional features can be added by other programmers easily

III. System Architecture

Two main softwares were used in this project: WordPress and BlueHost. WordPress is a web publishing software, and Bluehost is a web hosting service. The website this field session team developed was created on WordPress and hosted through Bluehost. This means Bluehost holds the database, all tables, and all the page templates with custom functions the team created. Bluehost was created to host WordPress sites, so they very easily pass information between each other. The actual site pages are created through WordPress. Figure 1 below shows the relationships of the system.

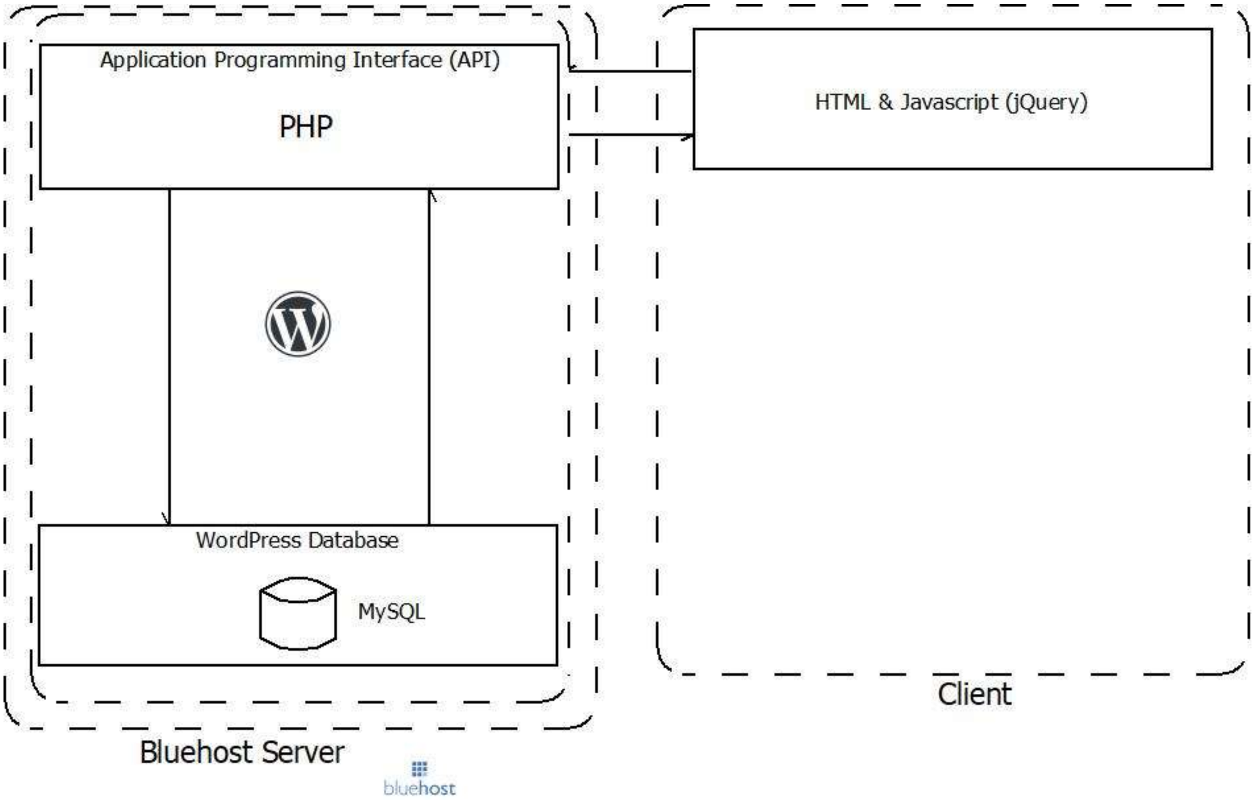

Application Programming Interface (API)
PHP
HTML & Javascript (jQuery)
WordPress Database
MySQL
Client
Bluehost Server
bluehost

Figure 1: Bluehost to WordPress structure

The system architecture design for this project centered around a clear and uncomplicated Client-Server design. This design was chosen as it allowed for the expansion of site by giving the client easy access to the backend of the website. However, the development of the site was tied to these two platforms, which will make it difficult should the client want to move the site off of either of these platforms later.

The diagram above provides an easily understandable view of the relationships between the server and client. The server is in charge of querying and retrieving data from the MySQL database. The Application Programming Interface (API) communicates with the MySQL databases. This is done by the client side. The client sends AJAX GET requests to the API via the Bluehost server. Information is received back as XML or JSON, which can then be displayed by the browser.

The architecture of this system was designed to utilize the server and tools that our client requested, while being easily maintained in the future. Websites created with WordPress include plugins and other features that help the administrator to manage and maintain the site. The plugins the team implemented are explained in detail in the User Manual, found in the Appendix.

IV. Technical Design

Within WordPress and Bluehost, this project was designed using a Model-View-Controller (MVC) framework to organize and separate sections of the code. The MVC framework is a common way to organize logic for a web application and makes visualizing, editing, and sharing the app much cleaner and easier. The model is made up of the database which, in a structured format, presents information to the user and makes decisions about how the website works dynamically. Templates holding HTML and CSS code make up the views which are what the user sees when interacting with the website. The user interacts with the view, which then sends information to the controller. The controller is JavaScript and PHP code that make up the logic that ties together the views and the model. It manipulates the database from the user interactions, which proceeds to update the view the user sees. A diagram of the project's MVC is shown in Figure 2 below.

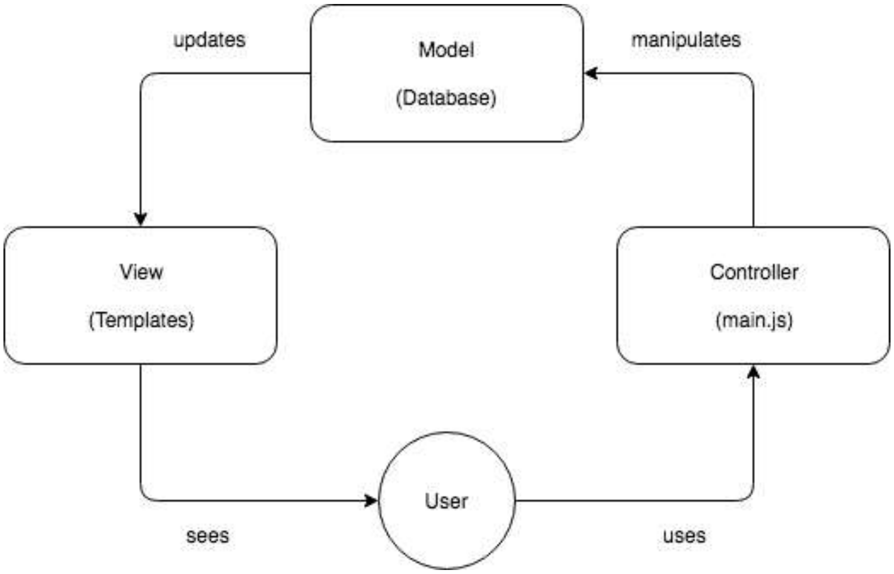


Figure 2: Model-View-Controller Interactions

The database design was one of the most important aspects of the project as the website was requested to be database-driven. The database design was redesigned/restructured several times to fit changing requirements of the project and improve efficiency of the site and the database. Figure 3 below represents the final database design for this project. When a user first comes to the site, they can create an Account. Accounts default to giving merely user privileges. However, Members can request that their privileges be changed by an administrator. When an Account is created, it also creates a new Member, which has more personal information about the user. Users must enter information about themselves and select certain options. This information is stored between these two tables in the database. A Member can register to attend a Conference. Conferences have many Events scheduled during them. Conferences also have many Sessions. Sessions contain the Publications that are presented at that specific Session. These relations, as well as the attributes that each table contains, are shown in Figure 3 below.

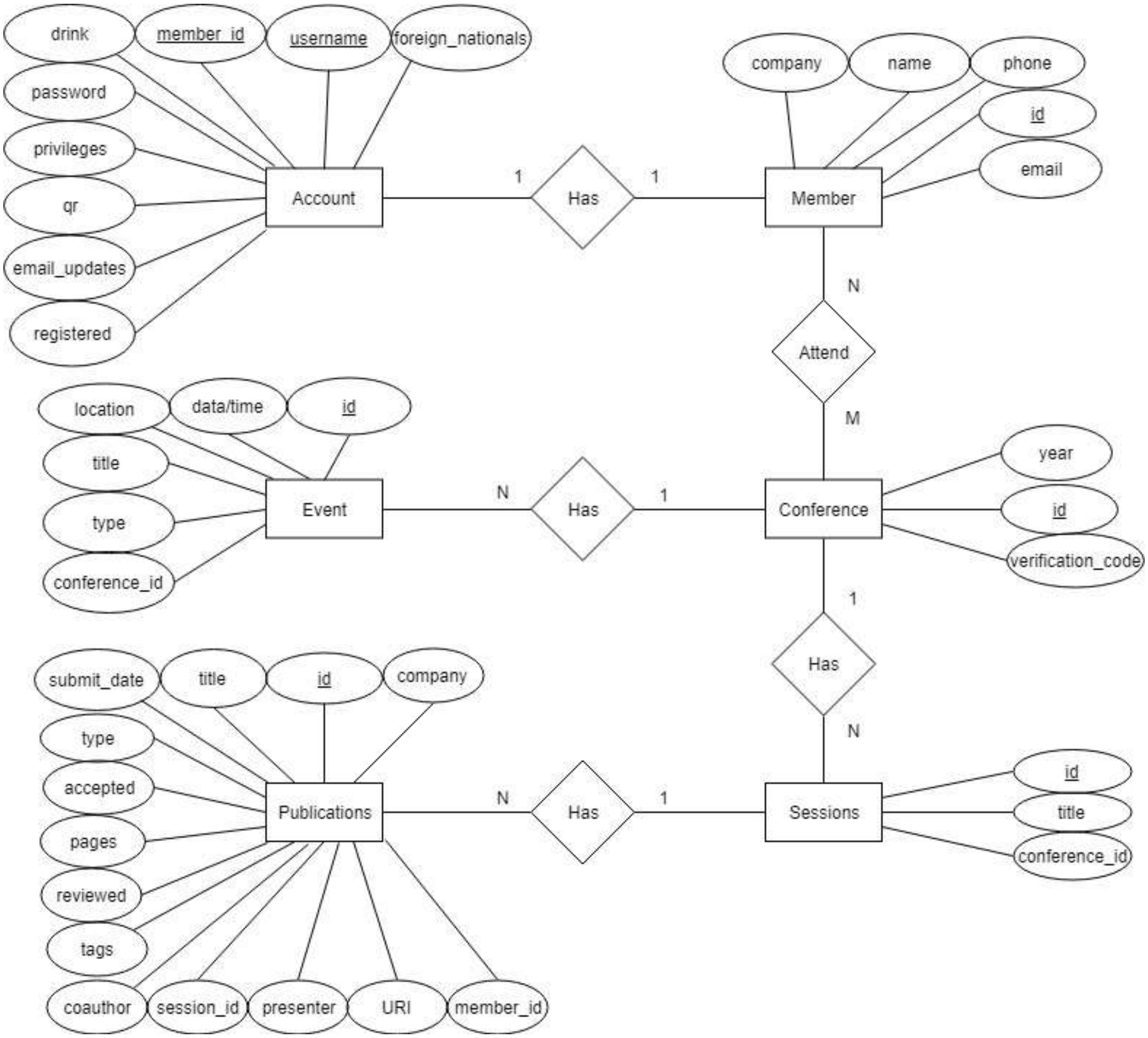


Figure 3: Final Database Design

This database design is normalized to the Third Normal Form (3NF) as there are no transitive functional dependencies in the database where an attribute determines every other attribute. For example, the *conference_id* determines what *year* and *verification_code* is, and it also determines which *Conference* the *Event* is held, but it doesn't determines also the other information of *Event*. Not only that, this normalized database allows no redundant data within the database thus improving the accuracy of result fetched from querying against the database with no repeated values.

V. Decisions Update

Throughout the six week process, the team made a number of design decisions to ensure creation of a quality product. From database design, website format decisions, to the hosting service used, each decision was carefully discussed and decided upon as a group with the final goals of the project in mind.

The primary design decision the team had to make in the context of this project was whether the site would be hosted through Amazon Web Services (AWS) or Bluehost. Initially the client requested AWS be used, so the team began the project working to understand AWS and all the services offered. It was challenging to understand how it worked, and which of its services were needed for the website. The client then requested a trade study document of the differences between the services of AWS and Bluehost, their current hosting site. The team analyzed the two primarily in terms of cost, usability, learning curve, documentation, scalability and compatibility. Through this research, the team ranked each of these categories, and determined that Bluehost would be a better host for this project. The current site is hosted through Bluehost, so the client will not have to transfer anything at the completion of the project. One of the biggest differences between the two was ease of use- AWS has a lot of functionality, but is difficult to understand because it is overloaded with capabilities. Bluehost, on the other hand, is strictly web hosting, and is already partnered with Wordpress, on which the site is created. Bluehost is easy to pick up and understand, which was vital for the team given the project timeline, while AWS has a slower learning curve. Beyond this, the two offer very similar services. The team presented the trade study to the team's client, and after reviewing the analysis, it was decided to move forward with Bluehost.

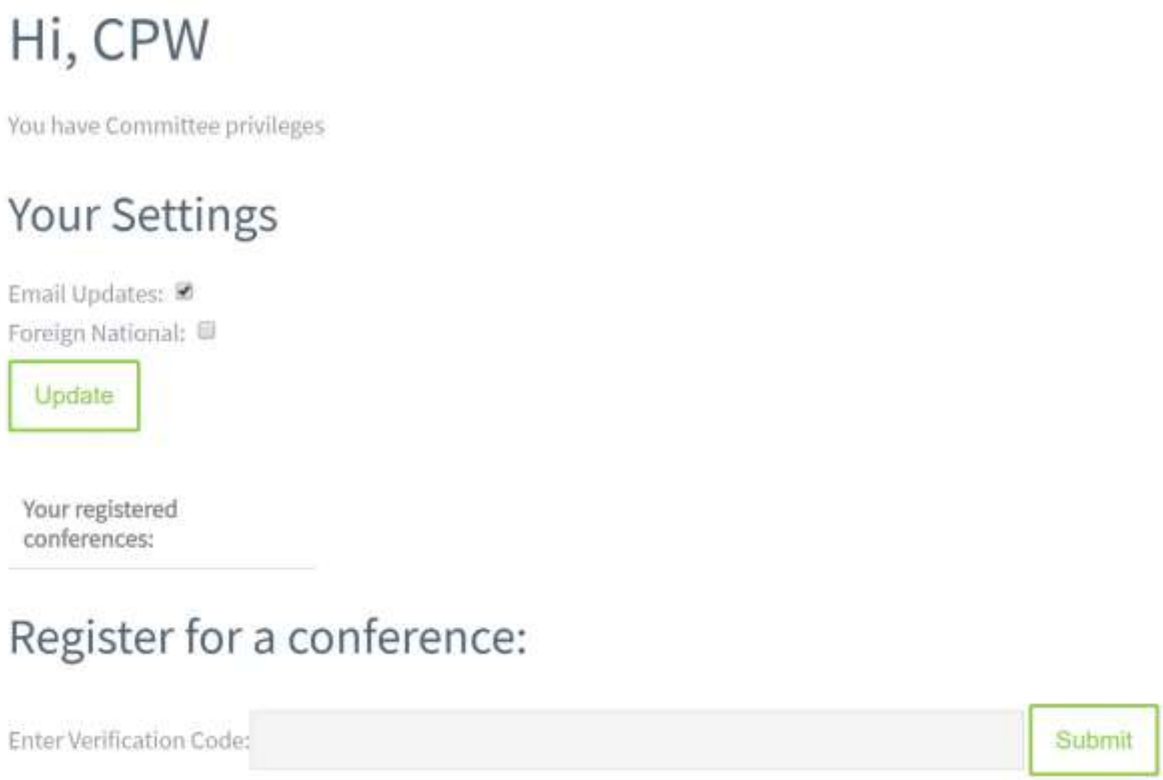
Another design decision the team made was in regards to the database design. Much of it was natural and fell into place easily. However, the team struggled to decide how to handle events that conference attendees would sign up for. From the initial database design with this functionality, it was a very circular approach and felt very awkward. The team's design included three cross-reference tables, which made it very difficult to manage. It was decided to redesign how the database handled events to make it a more natural approach that will greatly simplify the coding without losing functionality or organization. In the final design, there is only one cross-reference table required, far simplifying the process. The final database design is shown in Figure 3 in Section IV.

Another pertinent decision was how to present the data and information on the website. For some of the tables, the team found plugins that were already created. For the rest, code had to be written to customize the display. To decide which approach was required, the team researched different plugins available, and then read as much as was available online to understand how much functionality the plugin provided, and how customizable it was from there. If it was found to be a suitable plugin, it was integrated and then tested for edge cases. If no plugin was found to be comprehensive enough for the page’s needs, the team created a template and coded the page from scratch.

The rest of the design decisions have been in regards to the organization of the pages on the website. The team began with the design from the old website, and then looked at what new functionality the client wanted to restructure the site. A site flow map was created to show the client the vision for the new site, and feedback was obtained from our client. The team then adjusted the site layout from this feedback. The site map is shown in the user manual, in Section VII, Appendix.

VI. Results

The primary goal of this project was to turn the client’s current site into a database-driven website. The client gave the team an extensive list of features she hoped to integrate into their existing website to modernize it. The team spent the first three weeks setting up the working environments, researching about databases and learning PHP and HTML. They received account information in week three to create a WordPress site hosted through Bluehost. After understanding how these worked together and understanding what they would be able to complete in the time frame, the team rescoped the project, narrowing in on the list of most important requirements. The client’s highest priorities were: adding user accounts, user login functionality and publication submission and review functionality. These functions, and more, have been implemented into the WordPress site. Screenshots from the Account Page and the Review Publications Page are shown below. Both of these are from the view of a Committee Member. The Account page shows your user privileges, and allows you to register for new conferences and update user settings.



Hi, CPW

You have Committee privileges

Your Settings

Email Updates: ☑
Foreign National: □

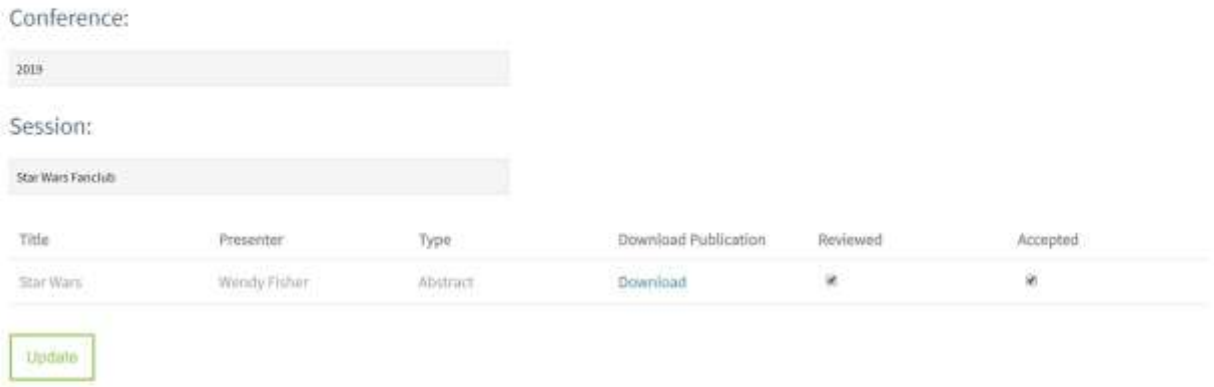
Your registered conferences:

Register for a conference:

Enter Verification Code:

Figure 4: Account Page for Committee Member

The Review Abstract page is only visible to Committee Members. Here, they can downloaded submitted abstracts to decide to accept them for the upcoming conference.



Conference:

2019

Session:

Star Wars Fanclub

Title	Presenter	Type	Download Publication	Reviewed	Accepted
Star Wars	Wendy Fisher	Abstract	Download	☑	☑

Figure 5: Review Abstract view for Committee Member

The team also created a user manual so another programmer can easily understand the framework they built to add additional features. This was very important to both the team and the client so the work can be continued after the field session project is completed. An abbreviated version of this manual is included in Section VII. Appendix. Some possible additional features include:

- Social interaction feeds
- QR code scanner
- Drink ticket counter
- Event RSVP functionality
- Email notifications
- Survey tool

The team tested the functionality and visibility of the site in a variety of browsers. The site is fully tested and functional from Chrome, Microsoft Edge, Safari, Internet Explorer and Firefox. The team also tested the functionality and visibility from a variety of mobile devices running different software.

The team learned a great deal throughout the six-week process. Working for a client presented a slew of new learning opportunities. The team learned to be flexible in their planning, creating a schedule that was detailed enough that they were working productively, but flexible enough that it was easy to adjust to changing requirements. The team also learned how to be more efficient in their communications with the client. Almost every day, they discovered an extensive list of questions built up. The team had to learn which of these they could make informed decisions on, and which of these needed client input. For those that needed client input, it was best to accrue the list of questions and then ask all at once either at the end of the day or at the start of the next workday. By doing this, the team found they often answered their own questions, or found that they didn’t need to be asked at all. The field session team also developed an Agile process that worked well for their team. They began each morning with a team scrum to bring everyone up to speed on the progress of the site and the highest priorities for the coming day. The team organized their tasks through Asana, a project management software. This kept them working on the highest priority tasks and ensuring forward progress.

At the start of the project, none of the team was very familiar with databases. However, through their research and work, the whole team has become much more comfortable working with databases. Everyone improved in their knowledge of writing PHP, JavaScript and HTML scripts. Another important lesson for the group was the initial planning and understanding of the project. They did not fully understand what software and platforms they would be using to develop this website at the start, which led to a lot of missed time in the first few weeks. Finally, the team greatly improved in their research and learning abilities. Because they had to learn how to implement all the functionality from scratch, the team had to learn how to learn the pertinent information and make new connections that aren't explicitly explained anywhere.

Overall, the team believes they created a product that the client will be satisfied with, and of which they are proud. The new site will provide a lot of value to the client and will add significant organization and efficiency to the American Astronautical Society.

VII. Appendix

Abbreviated User Manual

American Astronautical Society Rocky Mountain Section Website

This in-depth manual explains all the features implemented by the Colorado School of Mines field session team during Summer 2018. The website is currently hosted through Bluehost and created on WordPress. The team's primary responsibilities were to create user accounts and provide a way for users to submit publications for the conference. Since this project is expected to be expanded on in the future, in-depth explanations of how to modify the project are included along with descriptions of added features.

I. Functionality

New functionality created include: creation of new user accounts, user login, publication submission and publication review. The implementation of these are discussed in this section.

User Accounts

Users of the site can either create a new account or login to their existing account. When creating the account, usernames are checked against the current usernames in the database to ensure that no duplicates. The account is verified in two different places to ensure that the user is entering the proper information in the correct format. Once an account is created, that account is automatically granted User privilege, which can be changed later by the system administrator or by committee members. Users can access a settings page which allows them to change certain information about themselves that is attached to their account. Certain pages can only be accessed if a user is logged in, explained in the next section.

Privileges

The site uses three login privileges. These privileges and their accessible features are:

- User
  - View accepted abstracts
- Member
  - View accepted abstracts
  - Upload publications
  - Register for conferences
  - View reviewed papers that were presented at conferences they have registered for
- Committee
  - View all papers and abstracts

- Upload publications
- Review/accept publications

When not logged in, visitors to the site can view accepted abstracts.

Publication Submission Workflow

Members are able to upload abstracts, papers and posters for the upcoming conference via a submission form on the Submit Publication page. Until the abstract has been reviewed and accepted by a Committee member, it will not be visible when searched for.

Once an abstract has been uploaded, reviewers will be able to see it on the Review Publications page. They will be shown a table with all abstracts associated with the selected session and their current reviewed/accepted statuses. They may download the abstract and change its reviewed/accepted status by using the checkboxes on the table and submitting changes via the Update button. After the abstract has been reviewed and accepted, it will be visible for all users regardless of login status or permissions when searched for. An abstract may be unaccepted by unchecking the appropriate box on the Review Publications page again.

After their abstract has been accepted, the submitting member is able to upload the corresponding paper. Fields in the submission form are auto populated to ensure that only papers with corresponding accepted abstracts may be submitted. At this point, the accepted abstract will still be visible to the public, but the corresponding paper will not be visible until it has been reviewed.

Once a paper has been uploaded, reviewers will be able to see it on the Review Publications page. Papers will be shown in the same format as abstracts and their statuses may be changed in the same way. Papers will be automatically accepted. After a paper has been reviewed, it will be visible only to members who have registered for the conference it was presented at. Papers are to be marked as reviewed after they have been presented at the conference.

This workflow is shown visually in *Figure 6* below.

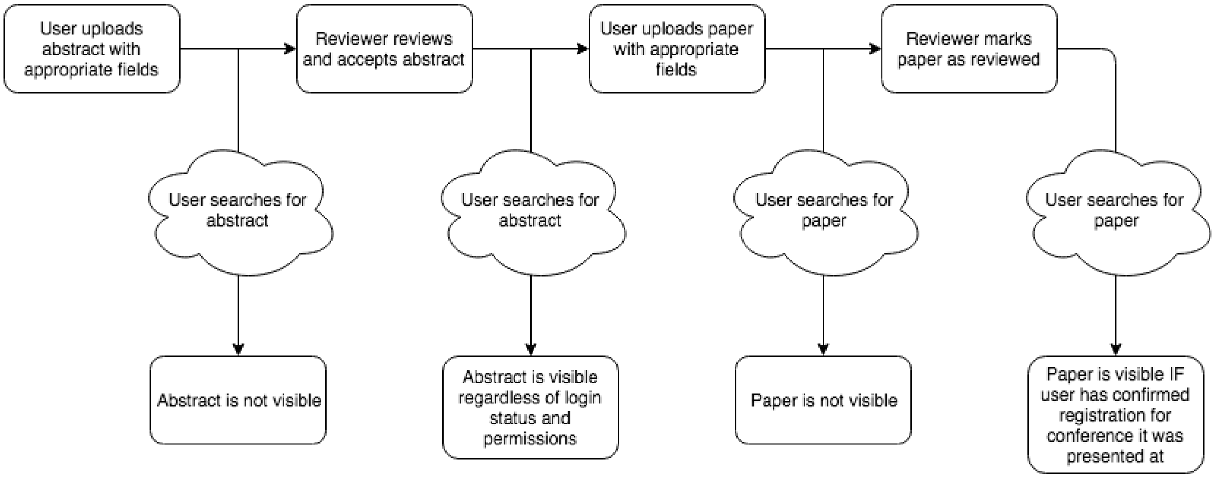

User uploads abstract with appropriate fields
Reviewer reviews and accepts abstract
User uploads paper with appropriate fields
Reviewer marks paper as reviewed
User searches for abstract
User searches for abstract
User searches for paper
User searches for paper
Abstract is not visible
Abstract is visible regardless of login status and permissions
Paper is not visible
Paper is visible IF user has confirmed registration for conference it was presented at

Figure 6: Publication submission workflow

Menu Structure

The website is organized to be as intuitive as possible. *Figure 7* below shows the site map layout. As shown, some pages are only accessible given certain privileges. The pink pages are the pages that are in the main menu of the site, with subpages shown in blue, and third level pages in green.

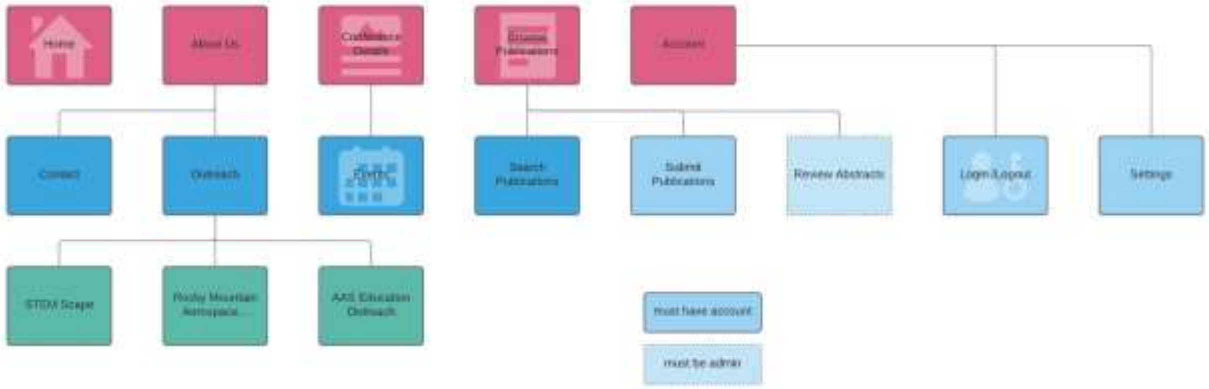


Figure 7: Website layout

II. Database

The database being used is the database automatically created by Wordpress for the site. This means that the custom database is combined with tables used by Wordpress.

Schema

The database schema used is shown below in *Figures 8 and 9*. Foreign keys are represented in the second figure by an arrow leading from the parent key to the borrowed key. The schema has gone through several alterations to represent data and relationships clearly. Currently, the only relationships represented are those used by the current site. If, for example, functionality was to be added which allowed members to sign up for events another foreign key relationship between the Member and Event tables will need to be added.

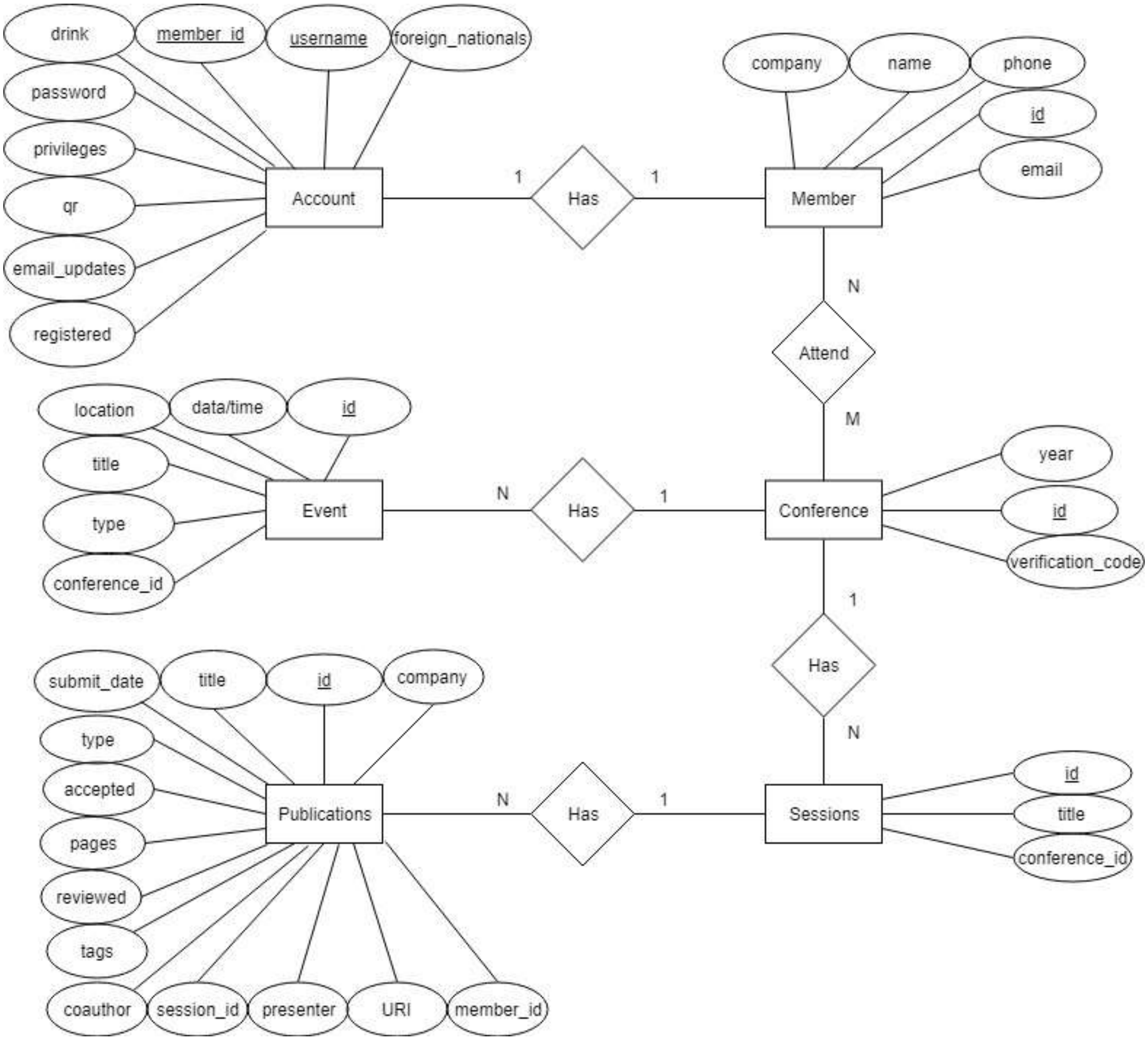
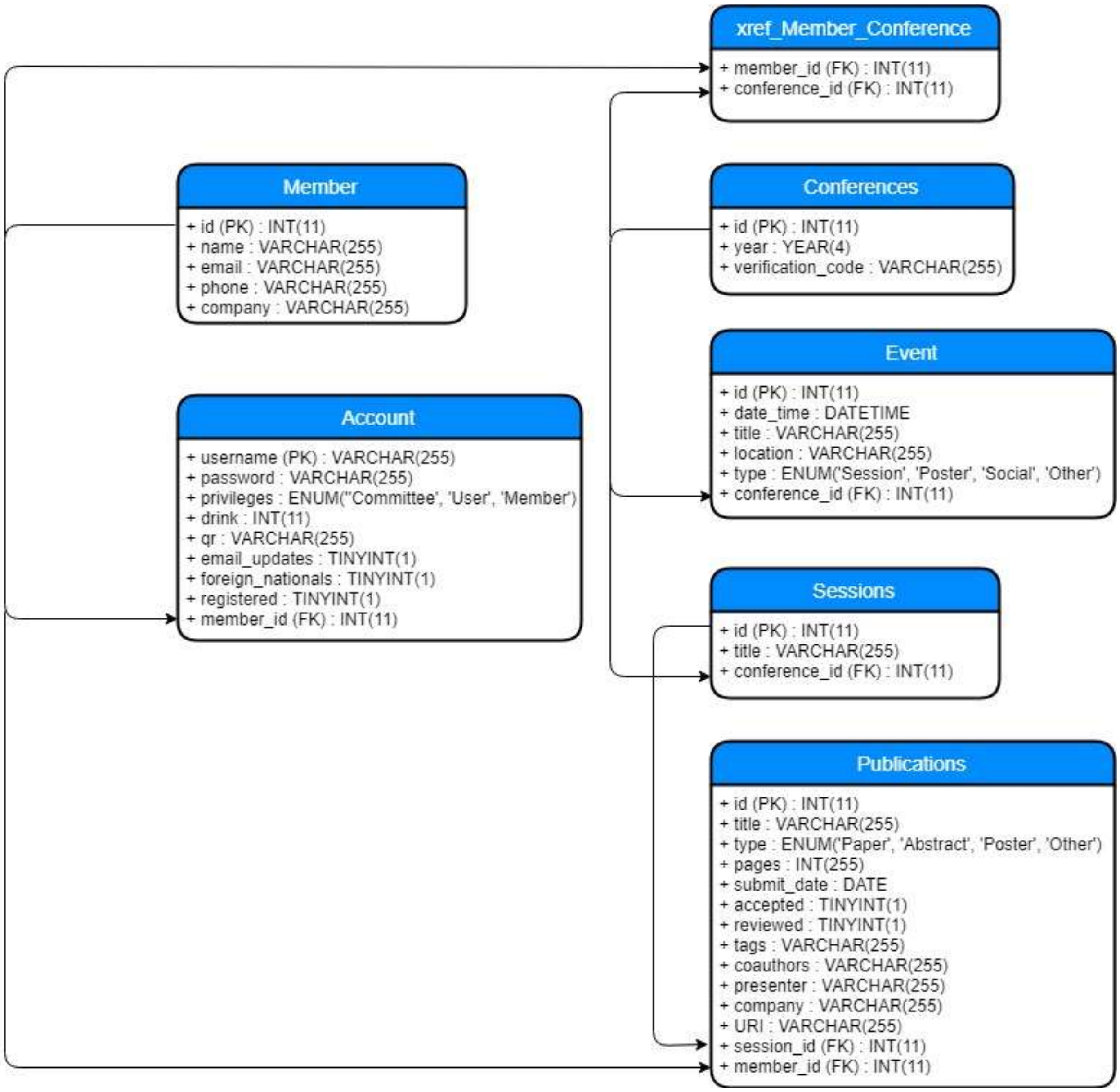


Figure 8: Database ERD



xref_Member_Conference
+ member_id (FK) : INT(11)
+ conference_id (FK) : INT(11)

Member
+ id (PK) : INT(11)
+ name : VARCHAR(255)
+ email : VARCHAR(255)
+ phone : VARCHAR(255)
+ company : VARCHAR(255)

Conferences
+ id (PK) : INT(11)
+ year : YEAR(4)
+ verification_code : VARCHAR(255)

Event
+ id (PK) : INT(11)
+ date_time : DATETIME
+ title : VARCHAR(255)
+ location : VARCHAR(255)
+ type : ENUM('Session', 'Poster', 'Social', 'Other')
+ conference_id (FK) : INT(11)

Account
+ username (PK) : VARCHAR(255)
+ password : VARCHAR(255)
+ privileges : ENUM("Committee', 'User', 'Member')
+ drink : INT(11)
+ qr : VARCHAR(255)
+ email_updates : TINYINT(1)
+ foreign_nationals : TINYINT(1)
+ registered : TINYINT(1)
+ member_id (FK) : INT(11)

Sessions
+ id (PK) : INT(11)
+ title : VARCHAR(255)
+ conference_id (FK) : INT(11)

Publications
+ id (PK) : INT(11)
+ title : VARCHAR(255)
+ type : ENUM('Paper', 'Abstract', 'Poster', 'Other')
+ pages : INT(255)
+ submit_date : DATE
+ accepted : TINYINT(1)
+ reviewed : TINYINT(1)
+ tags : VARCHAR(255)
+ coauthors : VARCHAR(255)
+ presenter : VARCHAR(255)
+ company : VARCHAR(255)
+ URI : VARCHAR(255)
+ session_id (FK) : INT(11)
+ member_id (FK) : INT(11)

Figure 9: Database Schema

PhpMyAdmin

PHPMyAdmin is a software tool included with Bluehost that allows the administrator to access the database tables and manipulate the data in those tables. This is the easiest place to change the structure of tables. It can be accessed through the cpanel of Bluehost once logged into hosting. PhpMyAdmin is in the Database Tools group on the cpanel.

Tabulate

Much like PhpMyAdmin, Tabulate allows for the ease of access to database records by the administrator. Tabulate is a plugin for WordPress, and as such is accessed through WordPress->plugins. Here, administrators can view the database tables and edit and add to them. CSV files can also be uploaded through Tabulate to populate the tables. Tabulate also provides shortcode to display auto-formatted tables on pages.

III. Uploading Data

CSV Files

The easiest way to upload CSV files into the database is to access Tabulate, which has built in functionality to allow for CSV files to be easily uploaded. However files, including CSV files, can also be uploaded through PhpMyAdmin. CSV files can be used to upload past publications.

Uploading Past Publications

The best way to upload past publications is through either PhpMyAdmin or Tabulate. To get a URI for these publications, the pdfs can be uploaded into the uploads folder and the URIs will be that path to the uploaded pdf.

Changing User Privileges

There is currently no streamlined functionality to change numerous user's privileges at once. Administrators can go into the Member table either in Tabulate or in the database through Bluehost, select a user by username (which will be their email address) and update their user privileges.

IV. File Structure

Most relevant files can be accessed in the theme folder. To view or update these files, first one must access the file manager on the Bluehost hosting cpanel. First login to Bluehost. Next click on the hosting tab and then cpanel. After gaining access to the cpanel and clicking on the File Manager icon, the control panel for the File Manager will appear. Currently the website is located at gnski.com/Z/Desktop. To move it to gnski.com (or the root of another domain, the files within the Desktop folder including index.php may be moved to the root (public_html) folder.

Templates

There are several templates that were created for the various pages in the site. Templates include HTML to be displayed on the page and are located in the theme folder.

JavaScript

There are two JavaScript files that are implemented the execution of the site which can be found generally in the same area as the templates in the File Manager. The JavaScript files contain functionality for templates, mainly making ajax requests to dynamically interact with the database. Changing information in any of the JavaScript files may require that changes be made in the functions.php file as well as the template file that uses any of the script.

Functions

All of the functions that are used in conjunction with the various templates are included in one PHP file that can be found in the File Manager as well. These functions implement much of the logic need to make the forms work properly and are where the majority of the code for the database interactions is contained. Some pages only require one function, while other pages require multiple different functions.

V. Wordpress Functionality

WordPress can be accessed by logging into Bluehost, accessing the cpanel, and then entering the WordPress administrator page through the MyInstalls icon on the cpanel. There are various WordPress plugins that are used by the site to function properly. Other WordPress plugins were used to make the site easier to use on the admin side, such as Tabulate, rather than through PhpMyAdmin that is provided with the BlueHost site.

Editing Pages

The majority of the WordPress pages are built with templates that are stored in the File Manager (please see file structure section for more information) which rely on logic found in separate PHP and JavaScript files that are in the Bluehost File Manager. To edit a page, the WordPress cpanel must be accessed, and then the page can be selected from the menu on the left. After the list of pages is displayed, the edit option will appear when hovering over the page. After entering the edit mode, the template can then be changed by selecting the appropriate template from the drop-down menu inside of "Page Attributes." The pages can also be edited using WordPress's built in editor, which is accessed on the same control panel screen in the large open space or Elementor, which can be used by clicking the large button titled "Edit with Elementor."

Plugins

AJAX File Upload

Ajax File Upload plugin is used on the Submit Publications page. This plugin allows users to upload a file from their computer to the site. The create_publications template adds the uploaded file to the database, along with other information about the publication as filled on in the corresponding form.

IfMenu

The IfMenu plugin was used to set visibility rules for certain pages of the website based on user privileges. Custom conditional logic for these privileges was made in the conditions function in functions.php. Custom and built-in visibility rules can be toggled on the Wordpress Dashboard->Appearance->Menus page.

Elementor

This plugin was used to format the static pages. Elementor gives a structured, elegant layout to the site. Pages may be edited with Elementor by going to a page and selecting “Edit with Elementor.”

Tabulate

This plugin allows for the administrator to interact with records in the database to be easily created, edited, updated, and deleted. Although functionality was implemented to let the user directly create some records in the database, it seemed important to include an easy way for committee members and the administrator to have this functionality. This plugin also allows for the tables to be searched by all the different values in the table and there are several search conditions that can be used. Tabulate also allows for CSV files to be uploaded into the database for future use.