Prepared for: Anaheim Ballet

Project Name: Anaheim Ballet Website

Prepared By: Chris Lagunas

Contributors: N/A

Design Planning Summary

The goal of this new design is to ensure customers and clients are able to easily access the information they need while also being introduced to current performances, classes, and company updates. The need for this project stems from the organization's existing website, which has become outdated and lacks the interactivity and modern aesthetics expected by today's users. This limits both audience engagement and online visibility. The updated solution aims to address these challenges by improving user experience, enhancing interactivity, increasing online reach, and simplifying the process for administrators to update events and class information.

To meet these objectives, the proposed solution is a clean, responsive web application developed using HTML, CSS, and JavaScript, with React serving as the primary framework. The site will feature dedicated sections for performances, events, contact forms, and a photo gallery to highlight the company's artistry and community involvement. This approach aligns with Anaheim Ballet's brand identity, promotes stronger engagement with visitors, and provides a maintainable and scalable foundation for future growth.

Overview of Design Concepts

The Anaheim Ballet website will function as an interactive platform that allows users to explore the company's offerings and stay informed about its latest activities. Visitors will be able to browse upcoming events, view photo and video galleries showcasing past performances, and gain insight into the company's performers and artistic style. The site will also feature a subscription option for weekly or monthly newsletters, keeping the community informed about upcoming shows, pop-up performances, and collaborative events with local organizations.

In addition, the website will include a section highlighting the ballet instructors, displaying their professional certifications and brief biographies to help prospective students become familiar with their teaching backgrounds and expertise. The major components of the website include the Events Page, Gallery Page, and Contact Page, which together create a seamless experience for users to connect with the Anaheim Ballet. These components work together to promote engagement, transparency, and accessibility, while maintaining a consistent, visually appealing design which will reflect the professionalism of this organization.

Flowcharts

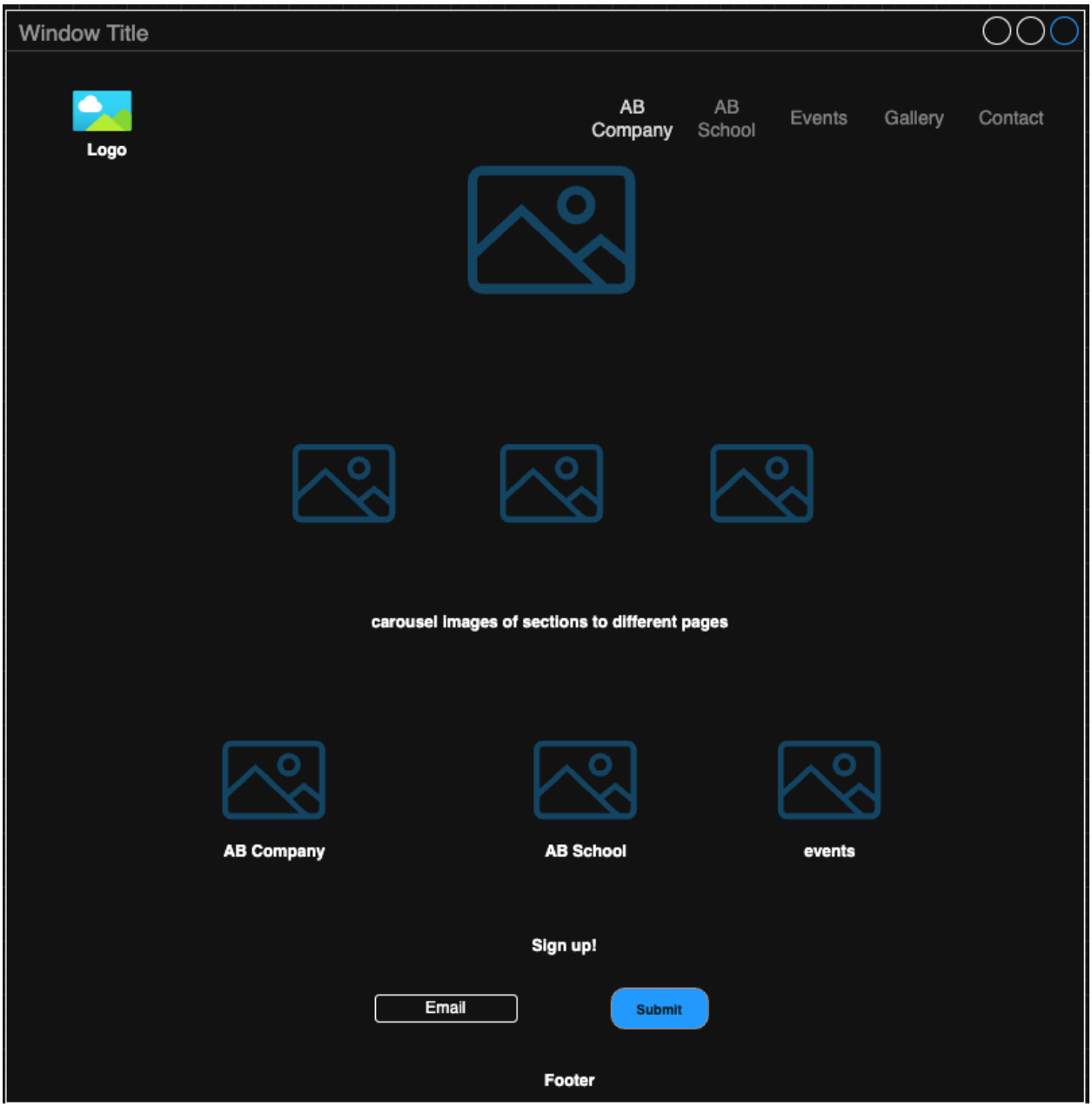

Figure 1 Index Page for the Website

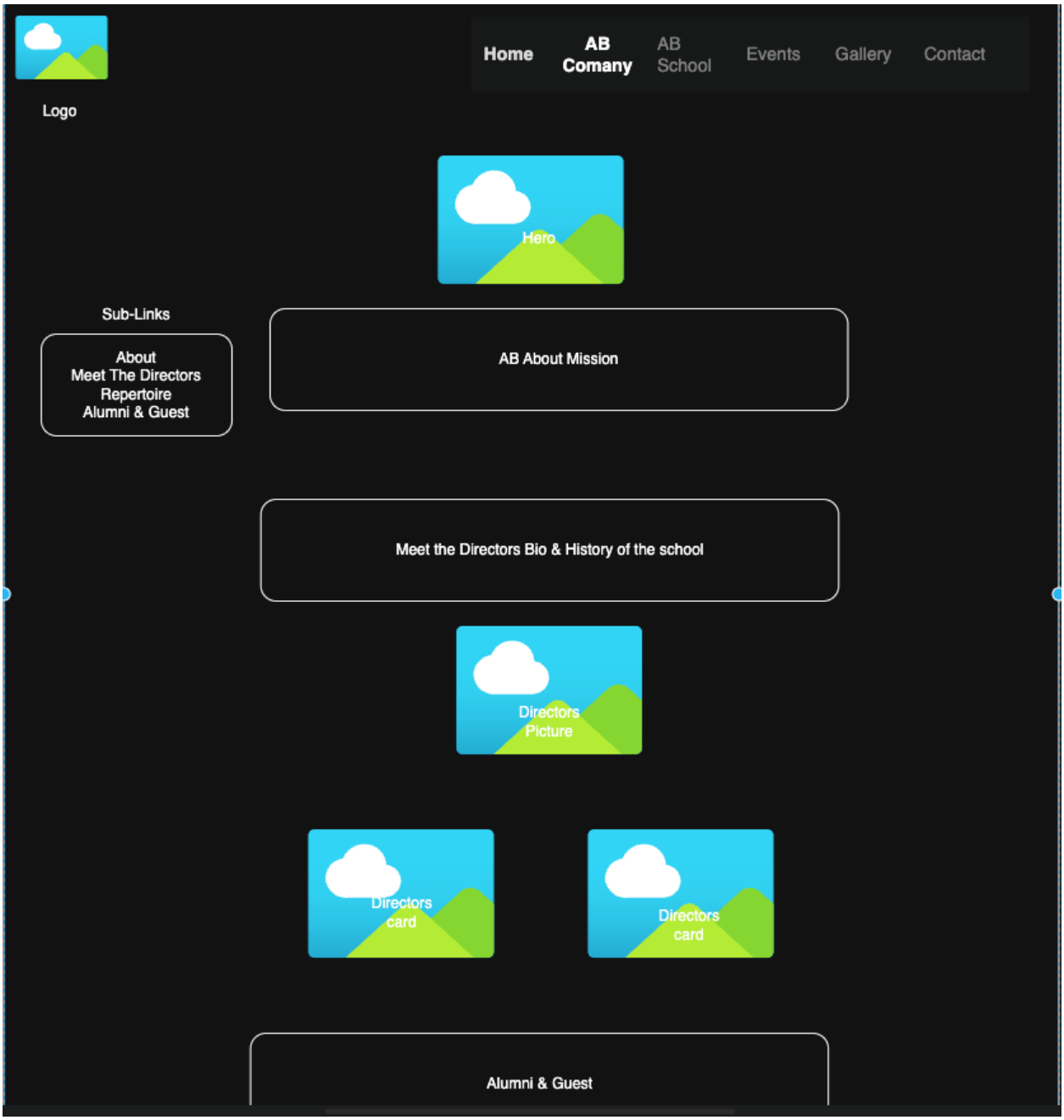

Figure 2 AB Company Page

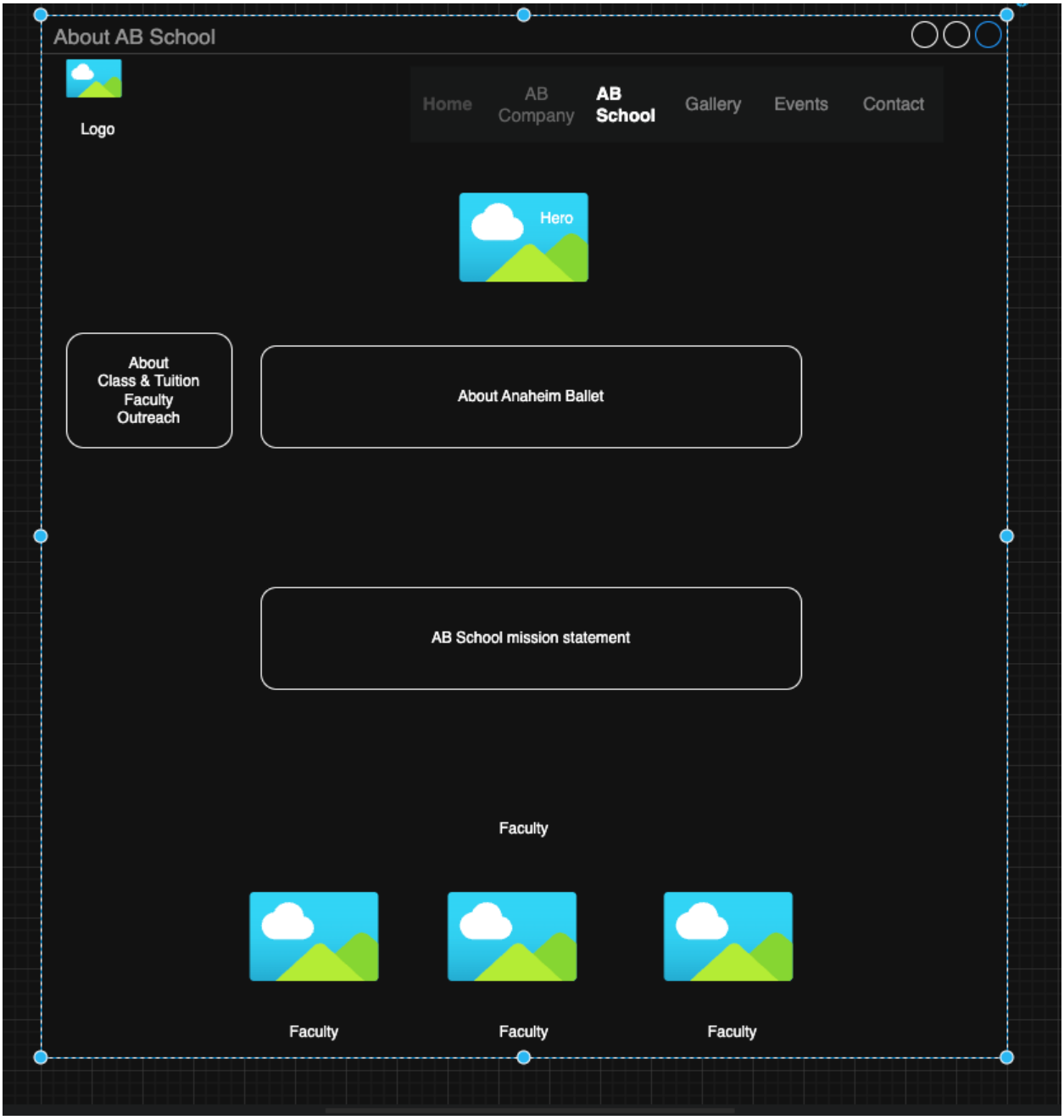

Figure 3 AB School pt.1

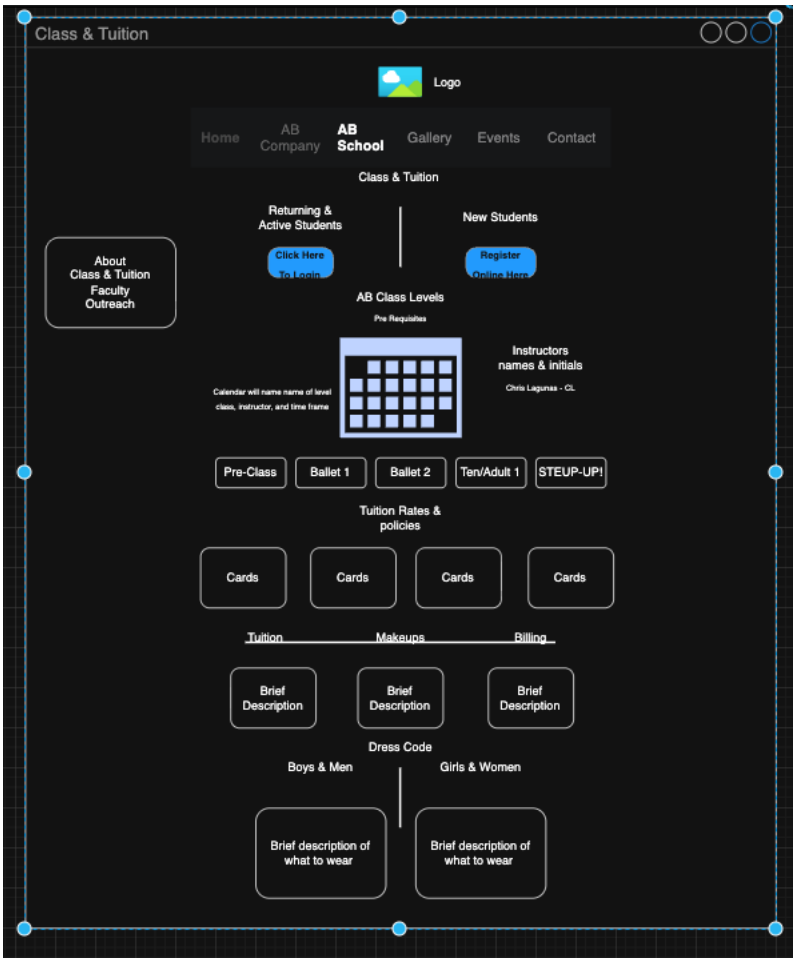

Figure 4 AB School Pt.2 Class & Tuition

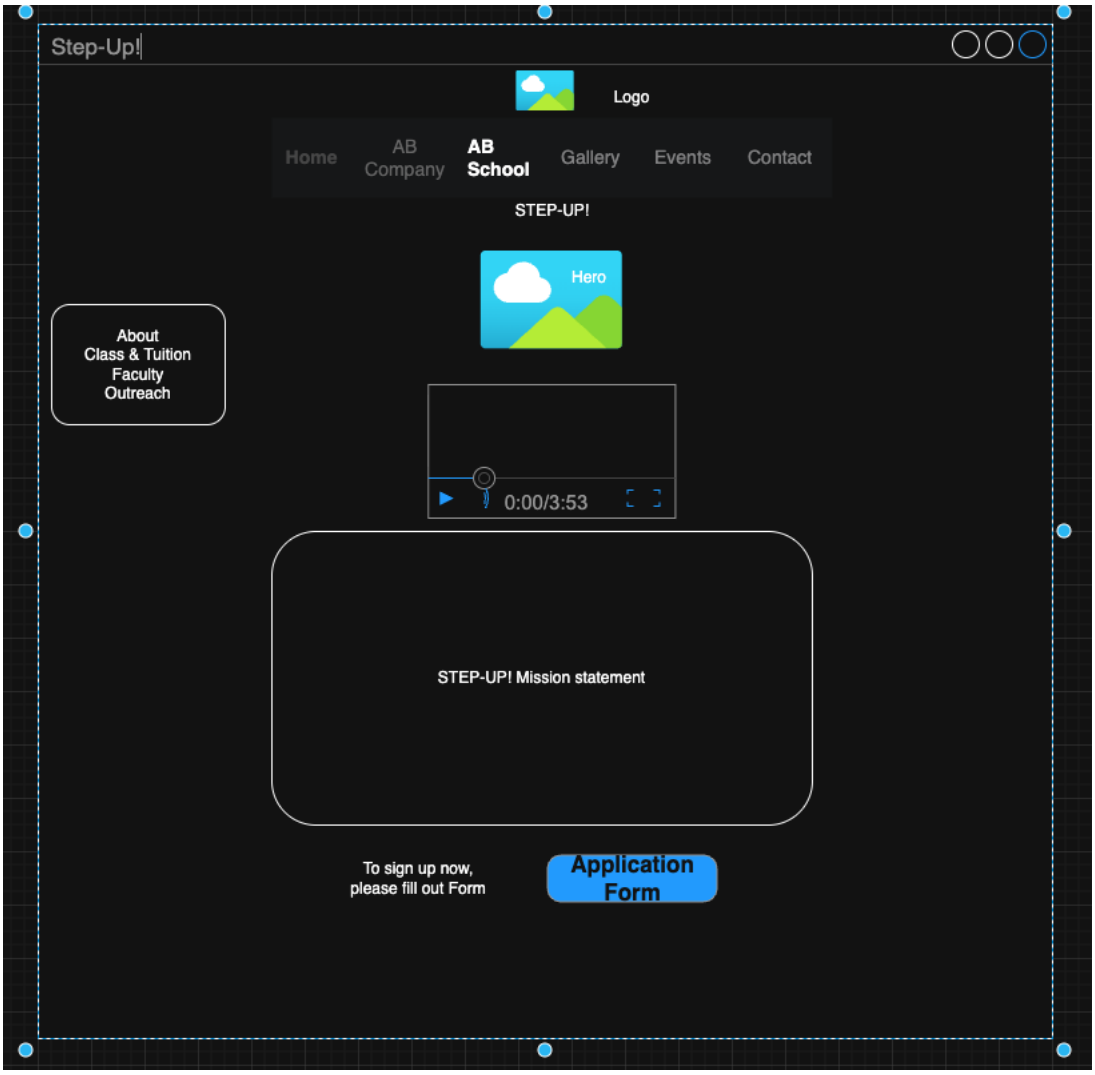

Figure 5 AB School pt.3 STEP-UP!

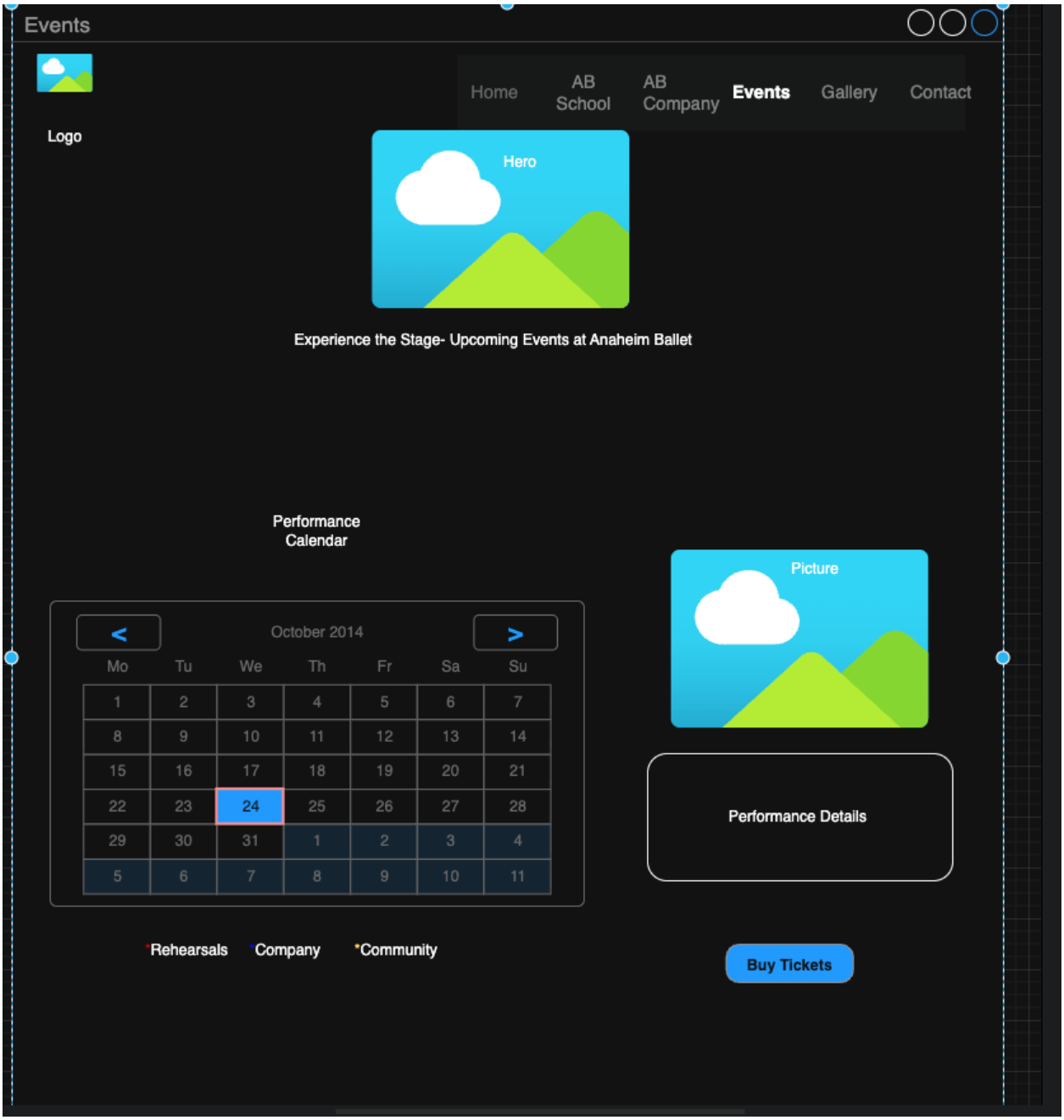

Figure 6 Events Page

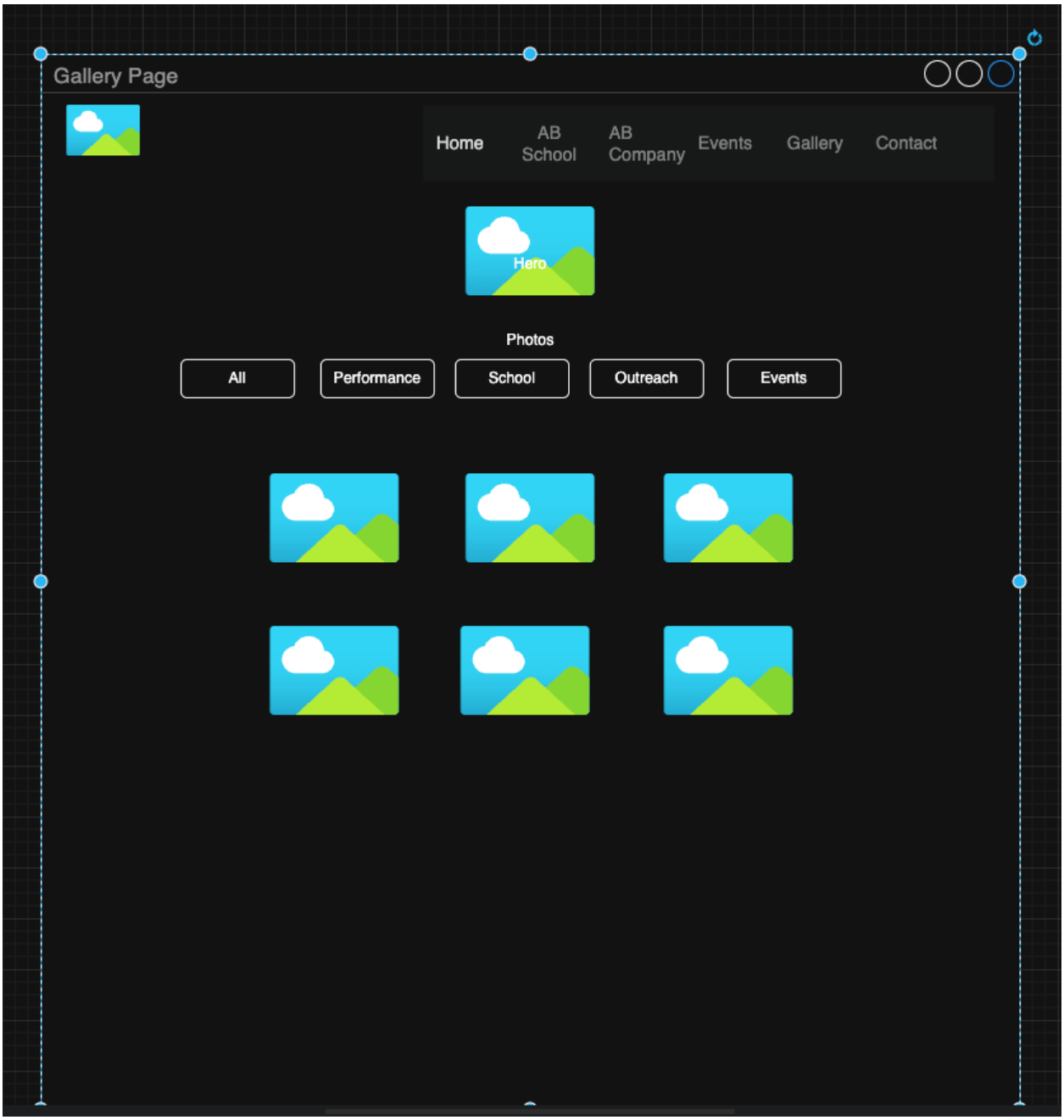

Figure 7 Gallery Page where viewers can switch between photos, videos, or meet the dancers with a name and a brief bio

Figure 8 Contact Page

Deliverable Acceptance Log

ID	Deliverable Description	Comments	Evaluator	Status	Date of Decision
1	Homepage UI Mockup	Accepted	Instructor	Complete	9/14/25

2	Gallery Photo Wireframe	Accepted	Instructor	Complete	9/14/25
3	Instructor Bio Section	Touched up	Instructor	Complete	9/14/25
4	System Flowchart	Touched up	Instructor	Complete	9/14/25
5	Final Architecture Plan	Submitted for grading	Instructor	complete	TBD

Detailed Solution Architecture

The Anaheim Ballet Capstone project follows a modern full-stack web application architecture using a client–server model. The system is divided into a React-based front end responsible for user interaction and presentation, and a Node.js/Express back end responsible for data processing, authentication, and database communication. The front end communicates with the back end through RESTful API endpoints over HTTP. Persistent data such as user accounts, newsletter subscriptions, and contact messages are stored in a MySQL database hosted locally via MAMP during development. This layered architecture promotes separation of concerns, scalability, and maintainability, allowing future enhancements such as administrative dashboards, role-based access, and expanded scheduling functionality.

The front end is developed using React and Vite, structured into reusable components and page-level views. Navigation is handled using React Router, enabling a single-page application (SPA) experience. Core pages include Home, AB Company, AB School, Class & Tuition, Events, Gallery, Outreach, Contact, Login, and Signup. Each page is composed of modular components such as carousels, calendars, forms, and profile cards. State management is handled locally using React hooks (useState, useEffect) for features such as calendar filtering, form submissions, and UI feedback. Styling is implemented using scoped CSS files aligned with a consistent ballet-inspired visual theme.

The back end is implemented using Node.js with the Express framework. It exposes RESTful API endpoints to support authentication, newsletter subscriptions, and contact form submissions. Middleware is used for JSON parsing, CORS handling, and environment variable configuration. Sensitive values such as database credentials and JWT secrets are stored securely using environment variables. The back end communicates with a MySQL database using the mysql2 library and connection pooling to ensure efficient database access.

ER Diagram:

Figure 9 ER Diagram of how everything will be connected

UML Diagram:

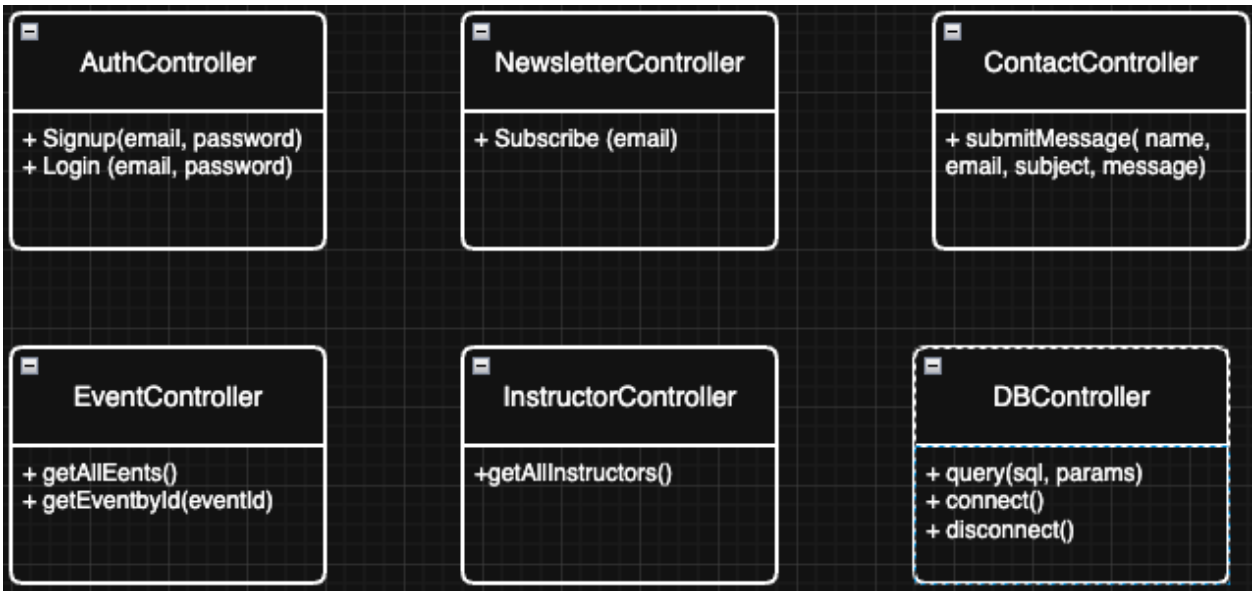

Figure 10 UML Diagrams that connect everything

Sequence/Workflow Diagram:

The sequence diagram illustrates how users interact with the Anaheim Ballet website and how data flows between the frontend, backend, and database. The steps as follows:

User Interaction:

A user visits the website via their browser and selects a page such as “Events” or “Gallery.”

Frontend Request:

The React frontend sends a REST API request (e.g., GET /api/events) to the Node/Express backend.

Backend Processing:

The Express server receives the request, validates parameters, and queries the MySQL database using the DatabaseService.

Database Query:

MySQL retrieves the requested information (e.g., event details, gallery items, or instructor bios).

Response Generation:

The backend formats the retrieved data as JSON and sends it back to the frontend.

Frontend Rendering:

React dynamically updates the user interface, displaying the data in event cards, gallery images, or instructor profiles.

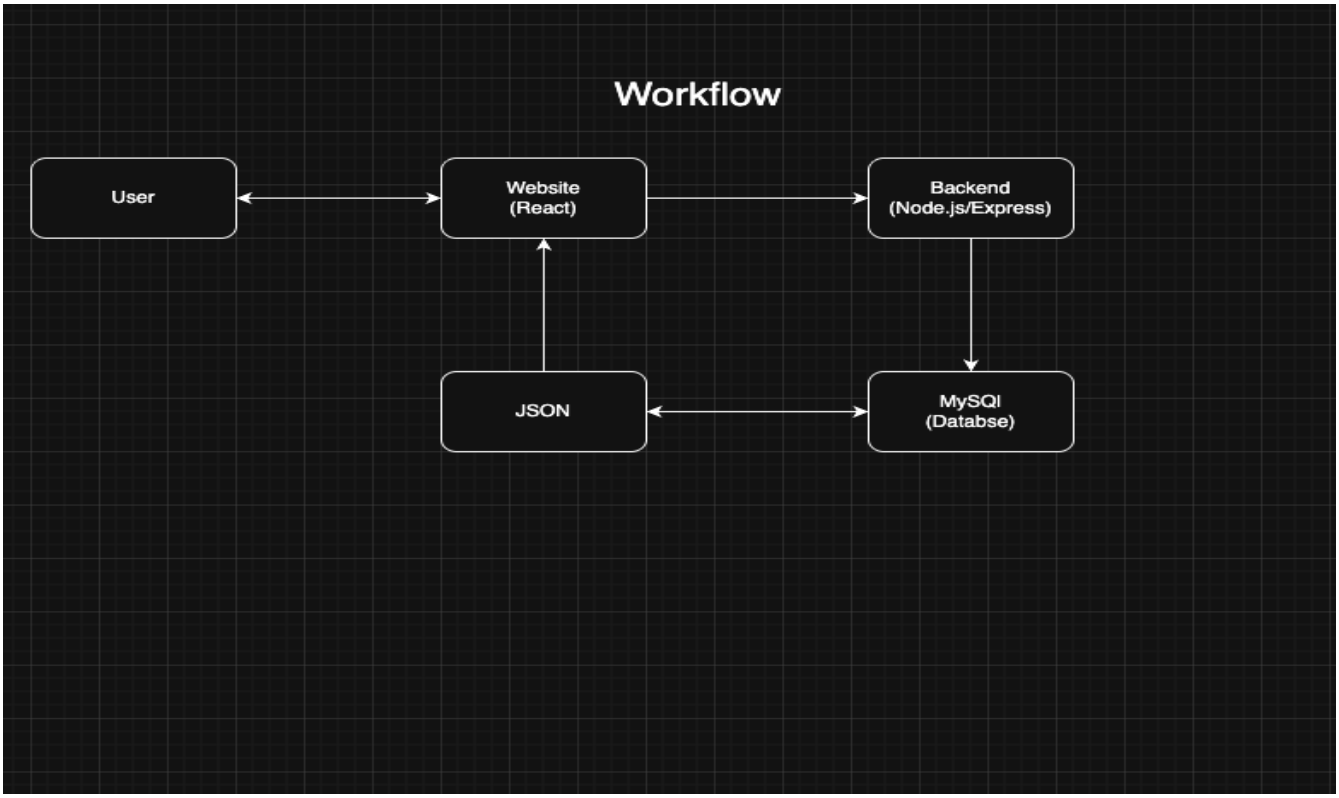

Figure 11 The Workflow

Algorithm Descriptions:

Event Display Algorithm:

User selects the “Events” page.

React triggers a GET /api/events call.

Backend validates the request and runs a SQL query:

SELECT * FROM events ORDER BY date DESC;

Results are serialized into JSON.

React renders the event cards with event title, date, time, and image.

Newsletter Signup Algorithm:

User enters an email in the newsletter form.

React sends a POST /API/newsletter request with the email address.

Backend validates the email format (Regex validation).

If valid, the email is saved in the newsletter_subscribers table with a timestamp.

Backend returns success message → frontend displays confirmation popup.

Contact Form Submission Algorithm:

User fills out the contact form and submits it.

Backend validates fields for empty or invalid input.

Data is stored in contact_messages.

Confirmation message is shown to the user.

Interface Specifications:

Home Page:

- Highlights upcoming performances, featured gallery images, and newsletter signup CTA.
- Responsive layout with hero banner and navigation to key sections.

Events Page:

- Displays a list/grid of upcoming and past performances.
- Each event card shows: *title, date, time, location,* and *Learn More* button.
- **Input/Output:** User clicks an event → fetches event details from /api/events/:id.

Gallery Page:

- Displays a masonry grid of photos and videos.
- Filter or toggle between “Photos” and “Videos.”

- **Input/Output:** On click, lightbox modal opens media file (image or embedded video).

Instructors Page:

- Cards for each instructor showing *photo, name, title, certifications,* and *bio excerpt*.
- **Input/Output:** User clicks instructor → optional detailed bio modal (future feature).

Contact Page:

- Includes form with fields: *Name, Email, Subject, Message*.
- **Input/Output:**
    - Input: user-provided data.
    - Output: success message on valid submission, error message on invalid data.

Newsletter Section:

- Simple email input form with a "Subscribe" button.
- Confirms with a success message upon submission.

Configuration Details:

For the developmental environment

Frontend: React, Vite, HTML, CSS, JavaScript

Backend: Node.js, Express

Database: MySQL (hosted locally via MAMP)

IDE: Visual Studio Code

Version Control: GitHub repository

OS: macOS with MAMP stack for testing

```
DB_HOST=localhost
DB_USER=root
DB_PASS=root
DB_NAME=anaheim_ballet
PORT=8080
```

Figure 12 Database will be used as

For deployment the following options are:

Local: MAMP environment for development and testing.

Future Hosting: Heroku or Vercel for public deployment.

Security:

The Anaheim Ballet website prioritizes user data protection and secure communication between clients and servers. These security implementations protect the site's integrity, safeguard user data, and ensure compliance with standard web security practices.

Implemented Measures:

- **HTTPS Communication:** All requests will use secure HTTPS connections (especially during deployment).
- **Input Validation:** Both frontend and backend validate form inputs to prevent SQL injection, XSS, and malformed data.
- **Environment Variables:** Database credentials and API keys are stored securely in .env files and excluded from GitHub via .gitignore.
- **Password Hashing (if login added):** Future admin login will use bcrypt for password encryption.
- **Access Control:** Only authorized administrators can update event, instructor, or gallery data.
- **Rate Limiting:** POST routes (Newsletter and Contact) limited to prevent spam or DDoS.
- **Regular Backups:** Database backups stored weekly to prevent data loss.

Hardware & Software Technologies

#	Technology	Description
1	VS Code	IDE for code development
2	Node.js + Express	Backend Server for handling requests
3	MySQL	Database for storing events, users, and gallery info
4	REACT	Front-End framework for the UI
5	MAMP	Local development environment for testing

Revision & Signoff Sheet

Date	Editor	Revision Notes
9/29/25	C.Lagunas	Initial Draft for review/Submission
10/3/25	C.Lagunas	Added ER/UML Diagrams
10/6/25	C.Lagunas	Added updated wireframes
10/7/25	C.Lagunas	Reviewed and updated acceptance log
10/10/25	C.Lagunas	Added final architecture details
10/12/25	C.Lagunas	Final Revision for Submission

Overall Instructor Feedback: