FIRST GROUP PROJECT REPORT

THE UNIVERSITY OF SYDNEY

P12 - Designing, developing and improving the website for BIOTech Future

Group Formal Tag: SOFT3888_TH10_Group1
Tutor Name: XINRUI ZHANG
Client Name: William Nixon, Daisey Weight, Joshua Aaron

Group members details

(indicate if any member is doing multiple units)

No.	Student Name	Student ID	Student Unikey
1	Hiolam Mui	530080868	hmui7674
2	Chenxi Zhang	530180264	czha0443
3	ZIYANG WANG	530257633	zwan9790
4	Qijun Zhou	510275976	qzho7932
5	Zixiao Cheng	530182531	zche4391
6	Junhao Fu	530804873	jufu0286
7	WENWEI LU	530031550	welu4128

Submission Date: 07.09.2025

Table of Contents

1 Introduction

The BIOTech Futures project is an initiative under the Faculty of Engineering that seeks to inspire and support the next generation of scientists, innovators, and entrepreneurs. As a platform, the BIOTech Futures website plays a central role in connecting high school students, mentors, and researchers, while showcasing opportunities for engagement, competitions, and outreach activities. However, the current website has several limitations in terms of usability, design consistency, and scalability, which restrict its effectiveness as both an educational and promotional tool.

To address these challenges, our project, Designing, Developing, and Improving the Website for BIOTech Futures, aims to redesign and rebuild the website with a modern, maintainable, and user-friendly architecture. The purpose of the project is threefold:

1. Enhance functionality and usability by improving navigation, interactivity, and responsiveness across devices.

2. Ensure visual consistency and branding through a cohesive design system that reflects the identity and values of BIOTech Futures.

3. Enable long-term sustainability by creating a toolkit and modular components that future developers can easily adapt and extend.

Our team adopts an iterative, prototype-driven development approach, integrating both client feedback and modern web development practices. By leveraging technologies such as React, JavaScript, and supporting libraries (e.g., react-router-dom, framer-motion, and lucide-react), the system is structured as a modular single-page application that is scalable and maintainable. This approach not only delivers immediate improvements in the website's presentation and usability but also ensures that the platform can evolve alongside the growing scope of BIOTech Futures.

Ultimately, the redesigned BIOTech Futures website will serve as an engaging, accessible, and professional online hub, strengthening its mission of promoting science communication and innovation among students and the broader community.

2 System Specification and Architecture Design

2.1 System Specification

2.1.1 Overview & Scope

The BIOTech Futures website is a React-based, modular single-page application that connects high school students, mentors, and researchers. The redesign focuses on usability, brand consistency, and long-term maintainability through a componentized architecture.

2.1.2 Users & Primary Use Cases

- Actors：students, mentors, supervi sors
- Use cases：discover programs & outreach, browse galleries, learn about the initiative, and contact the team.

2.1.3 Functional Requirements

- Global navigation & routing：Route to Home, About, Contact, and Gallery modes (Images/Posters/All).
- Homepage sections：Hero/Intro/Outreach/Timeline/Cards/Moments/ Footer are independently composed and revealed on scroll.
- Gallery：Unified base with mode switching, sorting/filtering, and lightbox, with consistent icons.
- Navigation UX：Top nav + fullscreen overlay menu with animations.

2.1.4 Non-Functional Requirements

- Usability & responsiveness across devices; visual consistency via a shared design system; sustainability/maintainability via modular SPA structure.
- Accessibility & performance targets(e.g., keyboard navigation, lazy assets).
- Animation behavior uses declarative patterns; reveal is driven by IntersectionObserver + framer-motion.

2.1.5 System Components & Responsibilities

- App Shell（main.tsx, App.jsx）：bootstraps React, sets up react-router-dom, declares route table; acts as a thin Router Facade.
- Pages（Home, About, Contact, Gallery*）：top-level views that compose sections and shared UI.
- Navigation Module（MainNav, Nav, MenuAnimation, LetterSlide, StyleInject）：global + gallery-specific nav with animated overlays.
- Homepage Sections：independent, reusable components styled by shared utilities.
- Gallery Module（GalleryBase + data/types/common）：strategy-like composition for Images/Posters modes with filters and lightbox.
- Shared UI & Utilities（common.tsx：COLORS, Container, useReveal, PageSection）：foundation primitives + reveal hook (Observer pattern).
- Assets（assets/images, Photo/*）：centralized static asset management.

2.1.6 Technology Stack & Libraries

- React SPA with react-router-dom（routing）、framer-motion（animations）、lucide-react（icons ）; IntersectionObserver for visibility.

2.1.7 Data & Content Handling

Images/posters are imported via centralized index and Photo/* folders; pages/components consume them without duplicating references.

2.1.8 Assumptions & Constraints

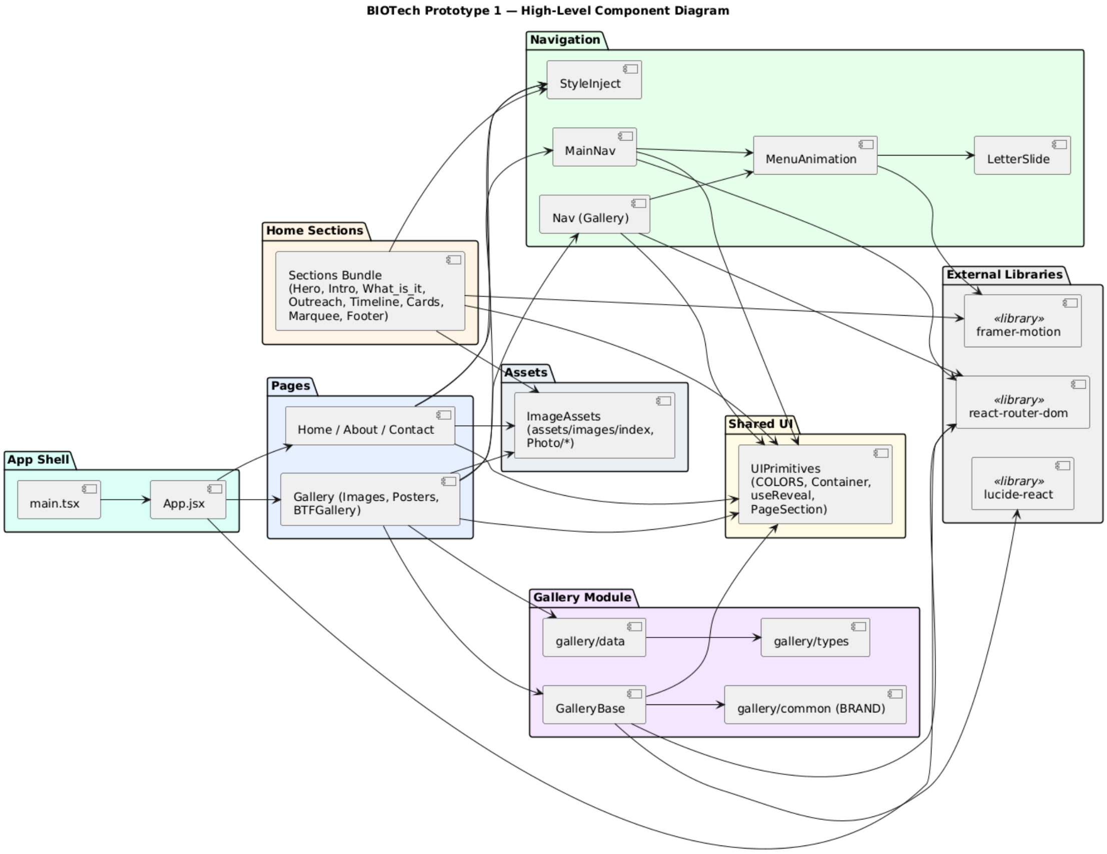
- Modern evergreen browsers; SPA navigation (no full page reloads).
- Architecture is evolving with iterative prototyping; diagrams/specs may refine over time.

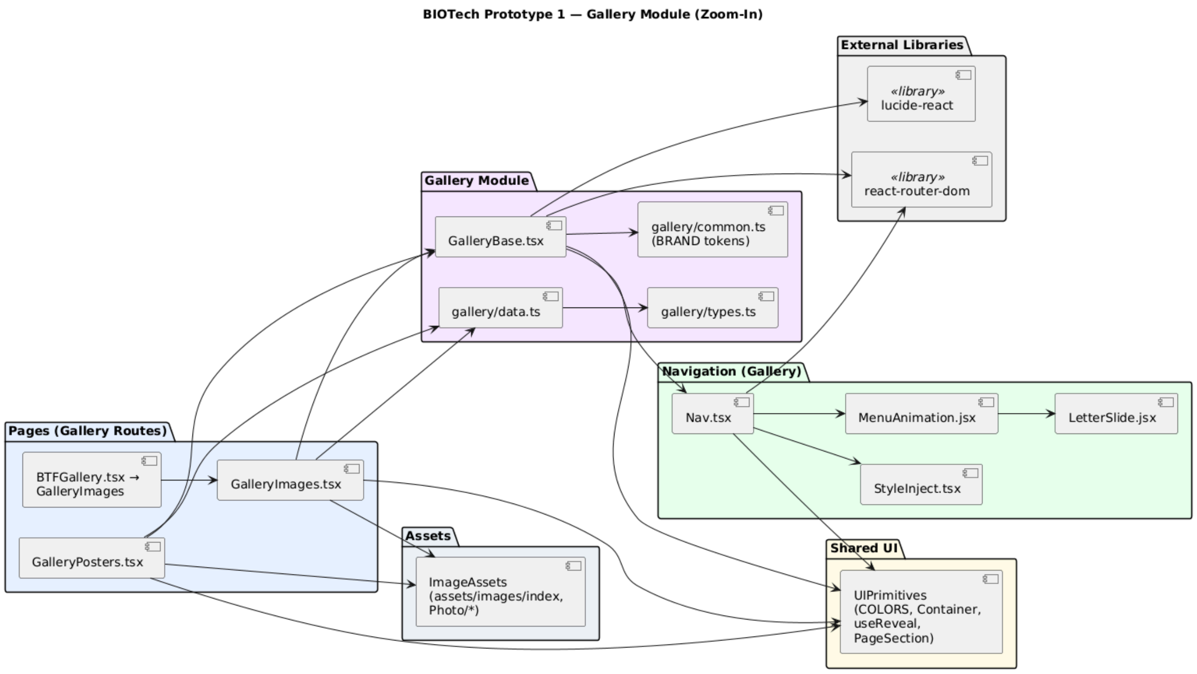
2.1.9 Quality, Testing & Metrics

- Unit/interaction tests for routing and gallery behaviors; manual checks for reveal timing and nav overlays.
- Success criteria: page load & interaction responsiveness, navigation clarity, and content discoverability aligned with project goals.

2.2 Architecture Design

2.2.1 Component Diagram


BIOTech Prototype 1 — High-Level Component Diagram
Navigation
StyleInject
MainNav
MenuAnimation
LetterSlide
Nav (Gallery)
Home Sections
Sections Bundle (Hero, Intro, What_is_it, Outreach, Timeline, Cards, Marquee, Footer)
External Libraries
«library» framer-motion
«library» react-router-dom
«library» lucide-react
Pages
Home / About / Contact
Gallery (Images, Posters, BTFGallery)
Assets
ImageAssets (assets/images/index, Photo/*)
Shared UI
UIPrimitives (COLORS, Container, useReveal, PageSection)
App Shell
main.tsx
App.jsx
Gallery Module
gallery/data
gallery/types
GalleryBase
gallery/common (BRAND)

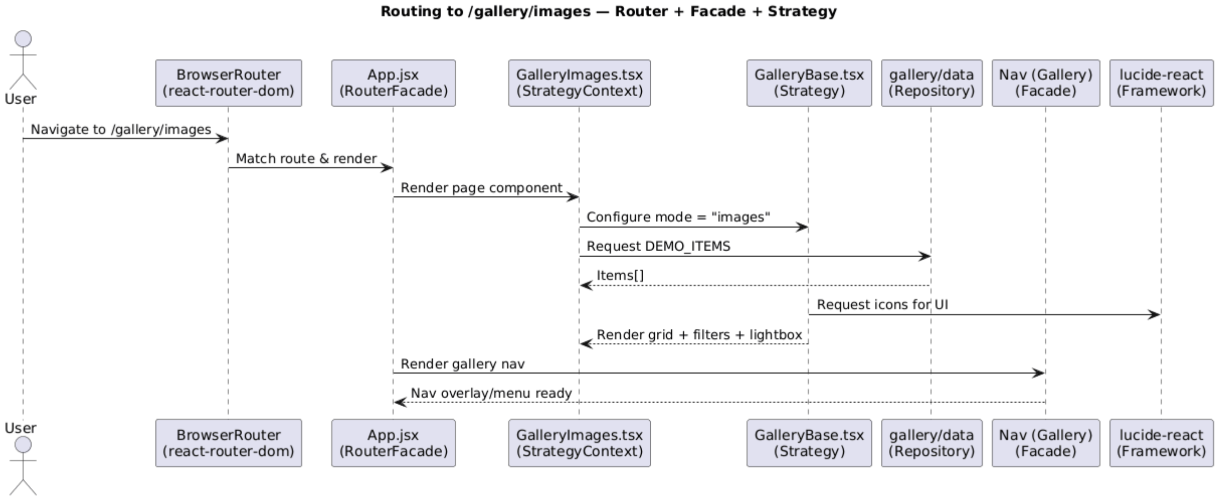


Remark

This project is being developed through an ongoing process of continuous development and integration. As such, the architecture, components, and their relationships are subject to refinement and change over time. The diagrams presented here reflect the current state of the prototype and may evolve as new features are implemented, existing modules are improved, or integration workflows are updated.

2.2.2 Design patterns & frameworks aligned with diagrams

Sequence diagram (1)



Description

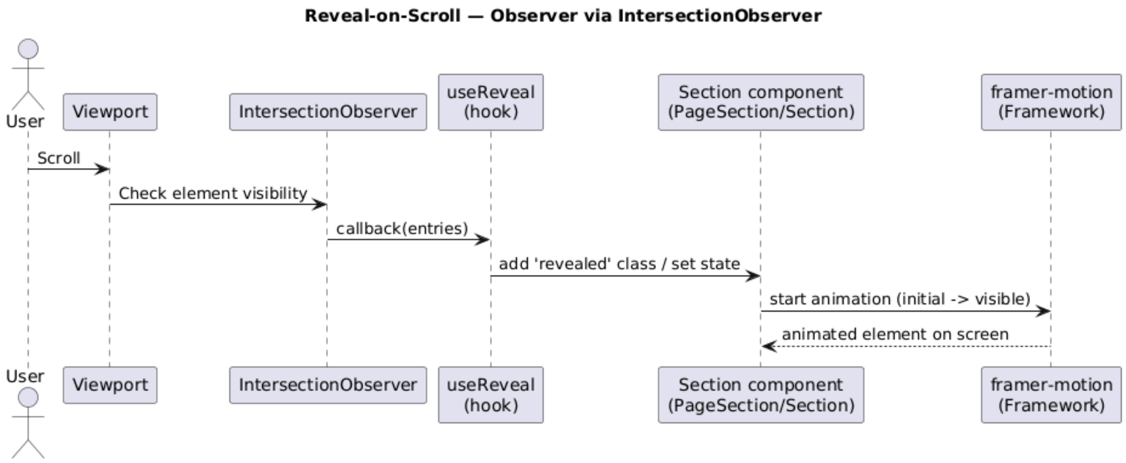
When the user navigates to the /gallery/images route, the BrowserRouter from react-router-dom matches the path and passes control to App.jsx, which acts as a Router Facade. App.jsx delegates rendering to GalleryImages.tsx, which selects the correct mode (“images”) and passes it to GalleryBase.tsx.

GalleryBase applies the Strategy pattern: it can switch between different gallery modes (images or posters) without changing its internal logic. The page component fetches items from gallery/data (acting like a simple Repository) and ensures type safety through gallery/types. For navigation, Nav.tsx (a Facade) renders the gallery menu and overlays, while lucide-react provides the iconography.

Pattern & Framework Highlights

- Router Facade: App.jsx hides routing complexity, using react-router-dom.
- Strategy: GalleryBase.tsx supports interchangeable modes (images, posters).
- Repository: gallery/data.ts supplies demo items consistently.
- Facade: Nav.tsx encapsulates menu logic for reuse.
- Frameworks: react-router-dom (routing), lucide-react (icons).

Sequence diagram (2)


Reveal-on-Scroll — Observer via IntersectionObserver
User
Viewport
IntersectionObserver
useReveal (hook)
Section component (PageSection/Section)
framer-motion (Framework)
Scroll
Check element visibility
callback(entries)
add 'revealed' class / set state
start animation (initial -> visible)
animated element on screen
User
Viewport
IntersectionObserver
useReveal (hook)
Section component (PageSection/Section)
framer-motion (Framework)

Description

As the user scrolls the page, the browser's IntersectionObserver API tracks whether specific section components are visible in the viewport. The useReveal hook registers these components as Observers, receiving callbacks when visibility changes. Once triggered, the hook updates the section's state, applying a "revealed" class. The section component then delegates animation to framer-motion, which smoothly transitions the element from hidden to visible.

Pattern & Framework Highlights

- Observer Pattern: useReveal implements a classic observer, where section components subscribe to visibility updates from IntersectionObserver.
- Hook as Mediator: useReveal mediates between the browser API and React state updates.
- Framework: framer-motion handles reveal animations declaratively.

2.2.3 How the system being designed and implemented

From a technical perspective, the system is being designed and implemented using an iterative, prototype-driven approach. In the first two weeks, we created three different prototypes, each with its own layout, styling conventions, and user interaction models. To guide our design choices, we studied a range of similar websites in the same domain and analyzed their strengths. For example, we examined how leading platforms handle navigation, gallery presentation, and content flow, then adopted and adapted the best elements to suit our client's requirements. This benchmarking step helped us ensure that our prototype aligned with modern design standards and offered an engaging user experience.

Once the client selected prototype01 as the base design, we moved into integration. The system architecture is organized around a React-based component structure, with a clear separation between the app shell (routing and bootstrap), page-level components (Home, Gallery, About, Contact), and shared UI primitives (navigation, styling utilities, container, and reveal-on-scroll). This modular design enables us to implement and refine features independently, while ensuring that all components can be integrated smoothly. We also leveraged off-the-shelf frameworks such as react-router-dom for routing, framer-motion for animations, and lucide-react for icons, which allowed us to focus on higher-level functionality rather than reinventing basic UI utilities.

In Weeks 3 and 4, our technical focus was on extending the homepage and improving the gallery subsystem. We implemented additional homepage sections as standalone React components (Hero, Intro, Timeline, Events, Cards, etc.) and styled them using a combination of reusable section wrappers and global style injectors. At the same time, we refactored the gallery into a Strategy-like component (GalleryBase) that supports multiple display modes (images, posters) with a shared core structure. This approach allowed us to integrate new gallery data sources and layouts in a flexible manner.

By Week 5, we expanded the scope of the implementation to include new subpages such as the About page and the Contact page. Each of these was designed following the same component composition pattern, ensuring consistency with the homepage in terms of layout, branding, and styling. Throughout development, we practiced continuous integration: merging updates from different developers into the shared repository, testing across routes, and ensuring that navigation, animations, and data bindings worked reliably. The result is a system that is not only visually modern, but also structurally robust and maintainable, with clear potential for further extension as client needs evolve.

2.2.4 Main technical components of the system

The system is built as a React-based single-page application (SPA), organized into modular and reusable components. Its design emphasizes separation of concerns, scalability, and maintainability. The following are the main technical components:

1. App Shell
    - Files: main.tsx, App.jsx
    - Provides the bootstrap layer for the application, mounting React to the DOM and setting up the routing context using react-router-dom.
    - Defines the global navigation structure by mapping routes to page-level components.

2. Pages
    - Files: HomePage.tsx, AboutPage.tsx, ContactPage.tsx, GalleryImages.tsx, GalleryPosters.tsx, BTFGallery.tsx
    - Represent top-level views in the system, each composed of multiple sections or shared UI elements.
    - Serve as entry points for rendering functional areas of the application (homepage, gallery, about, contact).

3. Navigation Module
    - Files: MainNav.tsx, Nav.tsx, MenuAnimation.jsx, LetterSlide.jsx, StyleInject.tsx

  - Implements a modern navigation experience, including both top navigation and a fullscreen overlay menu.
  - Uses framer-motion for smooth animation effects and StyleInject for cross-cutting styling concerns.

4. Homepage Sections
    - Files: Hero, Intro, What_is_it, OutreachEvents, Timeline, Cards, MomentsMarquee, Footer, and Section bundles
    - Each section is implemented as an independent React component for reusability and clarity.
    - Styled through shared utilities to maintain consistency across different sections of the homepage.

5. Gallery Module
    - Files: GalleryBase.tsx, gallery/data.ts, gallery/types.ts, gallery/common.ts
    - Implements the gallery feature using a Strategy-like pattern, where GalleryBase provides core logic and switching between image and poster views is handled by higher-level page components.
    - Supports sorting, filtering, and lightbox display; integrates icons from lucide-react.

6. Shared UI & Utilities
    - Files: common.tsx (COLORS, Container, useReveal, PageSection)
    - Provides reusable UI primitives (responsive container, consistent color palette) and utility hooks (e.g., useReveal which uses the Observer pattern via IntersectionObserver).
    - Acts as a foundation for consistency across all pages and sections.

7. Assets
    - Files: assets/images/index.ts, Photo/*
    - Centralized management of images and static resources.
      Allows components to import assets in a structured way without duplicating references.

8. External Frameworks and Libraries
    - react-router-dom: Routing between pages and subpages.
    - framer-motion: Declarative animation framework for smooth transitions and reveal effects.
    - lucide-react: Lightweight icon library for consistent, scalable vector icons.

2.2.5 Detailed explanation of how components interact

1) App shell → Pages (routing & bootstrap)

- main.tsx mounts React and wraps the app with BrowserRouter from react-router-dom.
    - Effect: all descendants can use <Routes>, <Route>, and <Link>.
- App.jsx declares the route table and renders page components:
    - / → HomePage.tsx
    - /gallery/images → GalleryImages.tsx
    - /gallery/posters → GalleryPosters.tsx
    - /gallery/all → BTFGallery.tsx (re-exports images)
    - /about → AboutPage.tsx, /contact → ContactPage.tsx

- Interaction pattern: App shell is a thin Router Facade—it never owns page logic; it just composes pages and passes control based on the URL.

2) Pages ↔ Navigation (shared and gallery-specific nav)

- HomePage.tsx, AboutPage.tsx, ContactPage.tsx import components/MainNav.tsx for the top navigation and global overlay menu.
- Gallery routes use Nav.tsx (gallery-specific nav with the same look/feel) so gallery UI can expose gallery-only actions without polluting the global nav.
- Both navs depend on:
  - MenuAnimation.jsx → orchestrates the fullscreen overlay animation.
  - LetterSlide.jsx → renders animated text pieces for menu entries.
  - StyleInject.tsx → injects global CSS utilities (typography, spacing helpers) so pages get consistent styling without duplicating CSS.
- Interaction pattern: Pages compose nav; nav delegates visuals to MenuAnimation/LetterSlide and borrows global styles from StyleInject.

3) Homepage composition (sections, reveal behavior, animation)

- HomePage.tsx composes section components:
  1. Hero.tsx, Intro.tsx, What_is_it.tsx, OutreachEvents.tsx, Timeline.tsx, Cards.tsx, MomentsMarquee.tsx, Footer.tsx.
  2. Sections.tsx further bundles TheBriefSection, PrizesSection, AnnualReportSection to keep the file tree tidy.
- Shared primitives: Almost every section uses components/common.tsx:
  1. Container (responsive layout), PageSection (consistent section padding/margins), COLORS (palette), and useReveal (IntersectionObserver hook).
- Reveal-on-scroll flow:
  1. HomePage renders a section with a ref.
  2. useReveal registers that ref with IntersectionObserver.
  3. When the section enters the viewport, useReveal sets local state and toggles a revealed class/state.
  4. Sections hand off the animation to framer-motion, which transitions from initial → visible.
- Interaction pattern: Sections are stateless view components that subscribe to visibility changes via useReveal (Observer), and delegate the animation to framer-motion.

4) Gallery module (strategy-like composition, data/types/branding)

- Entry pages: GalleryImages.tsx, GalleryPosters.tsx, and BTFGallery.tsx (alias to images).
  - Each page selects a mode (images vs posters) and calls GalleryBase.tsx with that mode.
- GalleryBase.tsx centralizes the gallery UI:
  - Renders filter/sort controls, the grid, and a lightbox.
  - Imports gallery/common.ts (BRAND tokens unique to gallery styling).
  - Uses lucide-react for icons.
- Data + typing:
  - gallery/data.ts provides DEMO_ITEMS consumed by image/poster pages.
  - gallery/types.ts defines the item schema so grid, filters, and lightbox have compile-time safety.
- Navigation integration: GalleryBase uses Nav.tsx for gallery-specific navigation; both use react-router-dom to keep URL and state in sync.

- Interaction pattern: Pages act as Strategy Context, choosing the mode; GalleryBase is the Strategy host, reusing one core layout/logic with mode-specific content.

5) Shared UI & utilities (cross-cutting foundations)

- components/common.tsx is a UI facade for:
  - Container (layout grid, max width, gutters),
  - PageSection (consistent section shells),
  - COLORS (design tokens for non-gallery pages),
  - useReveal (Observer hook built on the browser's IntersectionObserver).
- Interaction pattern: Pages, navs, and sections depend on these primitives for layout and behavior consistency; this reduces duplication and guarantees visual coherence.

6) Assets (centralized image pipeline)

- assets/images/index.ts exports named images used across sections and pages (e.g., gallery backgrounds, hero images).
- Photo/* stores additional imagery referenced by sections.
- Interaction pattern: Components import by name from the index module, not by path, which makes refactors safe and keeps imports uniform.

7) External frameworks (where they plug in)

- react-router-dom
  - Used in main.tsx (to create the router context),
  - App.jsx (route table),
  - Nav.tsx/MainNav.tsx (links, active route styles),
  - GalleryBase.tsx (optional route-aware UI).
- framer-motion
  - Used in Hero.tsx and animated menus (MenuAnimation.jsx) and any section that reveals on scroll.
- lucide-react
  - Used primarily inside GalleryBase (sorting, filtering, lightbox controls) and can be reused in nav/buttons.

8) Typical runtime flows (end-to-end)

A. Route → Page → Nav → Sections

1. User navigates to /.
2. BrowserRouter (from react-router-dom) matches route; App.jsx renders HomePage.
3. HomePage composes MainNav + sections, and injects styles via StyleInject.
4. Each section uses Container/PageSection and useReveal for on-scroll animation (framer-motion).

B. Route → Gallery Page → GalleryBase → Data/Types/Icons

1. User navigates to /gallery/posters.
2. App.jsx renders GalleryPosters, which sets mode="posters" and calls GalleryBase.
3. GalleryBase pulls BRAND tokens, icons (lucide), and items (DEMO_ITEMS) and renders filter/sort, grid, and lightbox.
4. Nav.tsx provides the gallery header and overlay menu; URL changes stay in sync via react-router-dom.

9) Integration & change management

- Modularity allows teams to add a new section or subpage without touching routing internals—just register a route in App.jsx and use shared primitives.
- Consistency is enforced via StyleInject, Container, PageSection, and color tokens.
- Safety comes from gallery/types.ts; changes to item shape surface as type errors where they matter.
- Evolvability: swapping gallery "modes," adding filters, or changing animations doesn't break unrelated pages because responsibilities are encapsulated.

2.2.6 Clear statement of components

Developed by the Group:
Core structure and functionality were built in-house, including the App Shell (bootstrap, routing), Page Components (Home, About, Contact, Gallery), Navigation Module (MainNav, Nav, MenuAnimation, LetterSlide, StyleInject), Homepage Sections (Hero, Intro, Timeline, etc.), the Gallery Module (GalleryBase, data, types, common), Shared UI & Utilities (Container, PageSection, COLORS, useReveal), and Assets Management (images, photos).

Adopted/Integrated:
External frameworks and APIs provide essential functionality: react-router-dom (routing), framer-motion (animations), lucide-react (icons), and the IntersectionObserver API (for reveal-on-scroll).

3 Discipline knowledge and Tools

Use and Application of Discipline Knowledge

Our team applied discipline knowledge from Software Engineering and Software Development throughout the design and implementation of the BIOTech Futures website. With six members from a Software Engineering background and one from Software Development, the project benefited from a strong foundation in both theory and practical development skills.

- Software Engineering principles guided our approach to project planning, task allocation, and process management. XP practices such as small releases, continuous feedback, and collective ownership were used to structure our iterations and ensure progress was transparent.

- Technical implementation relied on our training in programming and system design. React + JavaScript were chosen as the primary technologies to provide flexibility, scalability, and maintainability. Team members applied their knowledge of component-based architecture and responsive design patterns to build consistent and reusable sections across the website.

- Graphic design research: In addition to technical knowledge, we conducted significant design-oriented research to improve the visual quality of our product. We reviewed creative and professional websites such as *Unseen* (*Unseen Studio® – Brand, Digital & Motion*, 2022)and *CreateAnything* (*Anything - AI App Builder*, n.d.)using them as inspiration for layout structure, animation effects, and interactive elements. This process applied our discipline knowledge of user-centred design and usability principles, ensuring that the final website was not only technically sound but also visually appealing and engaging.

- Problem-solving using discipline knowledge: We addressed issues such as layout inconsistencies, animation lag, and responsiveness by applying debugging methods, analysing cross-browser compatibility, and refining React component logic. These activities drew directly on our background in testing, diagnosis, and correction of errors, which are central to software engineering practice.

- Software Development expertise contributed practical perspectives on usability, accessibility, and deployment readiness. This complemented the engineering focus on planning and process, ensuring that the final product was both technically sound and user-centred.

- Documentation and reporting drew on our discipline training in communication and requirements management. Minutes, scope plans, and progress reports were consistently produced, ensuring alignment with both client expectations and academic requirements.

Tools Used to Build the Systems

To support the design and development of the BIOTech Futures website, our team used a combination of development tools, frameworks, and collaboration platforms. These tools were selected to match the project's requirements for flexibility, usability, and maintainability.

- Languages and Frameworks
  - React + JavaScript: Core technologies used for building the website. React enabled us to apply a component-based architecture, making sections reusable (e.g., Hero, Prize, Past Winners). JavaScript provided interactive behaviour, while JSX simplified UI construction.
  - CSS-in-JS (within React): Applied for styling, responsive layouts, and animations without relying on external styling libraries.

- Development Environment
  - Visual Studio Code (VS Code): Chosen as the main IDE for its wide support of React development, extensions, and debugging features.
  - npm (Node Package Manager): Used to manage dependencies and libraries required for React development.

- Source Code Management
  - Git + GitHub: Git was used for version control, while GitHub served as our repository for code hosting and collaboration. All changes were transparent in commit logs, ensuring accountability.

- Design and Inspiration Tools
  - Canva & Figma: Used occasionally for wireframing and visual prototyping, supporting communication of design ideas before implementation.

- Collaboration and Communication
  - Slack: Main communication platform for coordinating daily tasks, sharing updates, and tracking issues.
  - Google Docs: Used to maintain weekly task logs, prepare meeting agendas, and record minutes.
  - Zoom: Used for client meetings and team discussions.

- Database Tools

- No live database was implemented, but frontend form will be designed in later stages. This ensured the system could scale with minimal rework.

4 Group processes

For the development setup, our team used Visual Studio Code as the main integrated development environment (IDE), with the project structured as a React + JavaScript application. We managed packages and dependencies using npm and version control was maintained through Git and GitHub, allowing us to create feature branches, push updates, and review code collaboratively before merging.

In terms of task management, XP emphasizes small releases and frequent iterations. We structured our work into short development cycles, where each cycle delivered working features such as the redesign style,restructure section and homepage development . Progress was tracked through regular client meetings and group meetings, and feedback was immediately incorporated based on both team discussions and client input. Our group contract defined expectations for participation, communication, and workload distribution, which all members adhered to.

Task Allocation

Task allocation was distributed based on team members' strengths and roles.

Week 1

In Week 1, our first task was to finalize the team composition and confirm the roles of each member. Once our group was established, we reviewed the list of available projects and collectively decided to work on the rebuild of the BIOTech Futures website.

We chose this project because it combined both technical and social impact. From a technical perspective, it allowed us to apply and extend our knowledge in React, JavaScript, and modern web design practices, while also experimenting with interactive features that improve user engagement. From a broader perspective, the BIOTech Futures initiative supports high school students, researchers, and mentors in science communication and collaboration. Rebuilding their website presented an opportunity to make the platform more accessible, visually appealing, and functional for its target users.

This alignment between our team's technical interests and the meaningful goals of the project made BIOTech Futures the most suitable and motivating choice

Week 2

In Week 2, during our tutorial session, we focused on clarifying the scope of our project. As a team, we discussed the main functions and features that would form the foundation of the BIOTech Futures website rebuild, and outlined the boundaries of what could realistically be achieved within the semester.

We also scheduled our first client meeting with the BIOTech Futures stakeholders. In preparation, we collaboratively developed a list of key questions to ask the client. These questions aimed to identify their priorities, gather feedback on the current website's limitations, and understand their expectations for improvements in usability, design, and functionality.

Evidence on Client interactions -emai Week 2l:
https://drive.google.com/drive/folders/14QTNHVXRAncBlsWgl_EnV71igAvo_QWG?usp=sharing

Evidence on first client meeting Zoom chat history
https://us02web.zoom.us/launch/chat/v2/eyJzaWQiOiJ3ZWJfc2NoXzZkNzY1ZjVhY2U3ZDQ1OGE4MDlmNWNjMzBkZjY4ZjFlQGNvbmZlcmVuY2UueG1wcC56b29tLnVzIn0

Week 2 Task allocation

- Bruce constructed new timeline for client
- Zixiao Cheng completed scope plan
- Ally completed Client minutes
- Qijun Zhou complete team minutes

Tasks allocation evidence can be found on the attached link:
https://docs.google.com/document/d/14uMgoV7fkWOzxIicN-Z6XCJrWxSOWGTr5_ghlnk16m0/edit?usp=sharing

Week 3

At the beginning of Week 3, after our first client meeting in Week 2, we realized that the client's expectations were different from the original project description provided in the project list. The initial description stated that the website rebuild should be done using Wix, but during the meeting the client clarified that they actually wanted the site to be rebuilt using Java technologies. This shift significantly affected our planning.

In addition, the client's requirements were expressed in a rather vague manner, as they were not entirely sure about the design style or the specific features they wanted. To respond to this uncertainty, our team decided to split into four subgroups, with each subgroup tasked to create a prototype. This approach allowed us to experiment with different layouts, styles, and interaction ideas, which we could later present to the client for feedback.

Week 3 Task allocation

Ally Mui

- Responsible for client liaison and meeting organization, including scheduling meetings and preparing discussion questions.
- Contributed to Group 2's prototype, focusing on the homepage design and layout to present an alternative style to the client.

Bradley Zhang

- Contributed to Group 2's prototype, focusing on the homepage design and layout to present an alternative style to the client.

Bruce Wang

- Contributed to Group1's prototype, focusing on the homepage design and layout to present an alternative style to the client.
- Worked on the gallery design, ensuring it aligned with the client's requirements and integrated smoothly into the overall website.

Zixiao Cheng

- Contributed to Group1's prototype, focusing on the homepage design and layout to present an alternative style to the client.
- Worked on the gallery design, ensuring it aligned with the client's requirements and integrated smoothly into the overall website.

Junhao Fu

- Contributed to Group 4's prototype, focusing on the homepage design and layout to present an alternative style to the client.

Kevin Lu

- Contributed to Group 3's prototype, focusing on the homepage design and layout to present an alternative style to the client.

Qijun Zhou

- Took client meeting minutes and two team meeting minutes, ensuring clear documentation of discussions and decisions.

Evidence on Client interactions -email week 3:
https://drive.google.com/drive/folders/1p6rbwar2A5ywpnMBnbm37-PcddpO7xb9?usp=sharing

Evidence on week 3 client meeting Zoom chat history:
https://us02web.zoom.us/launch/chat/v2/eyJzaWQiOiJ3ZWJfc2NoXzZkNzY1ZjVhY2U3ZDQ1OGE4MDlmNWNjMzBkZjY4ZjFlQGNvbmZlcmVuY2UueG1wcC56b29tLnVzIn0

Evidence on prototype and gallery design:
https://drive.google.com/drive/folders/1Jspzg6-95fRKNEiLflKNjXPfwx5r_rUm?usp=drive_link

Client feedback:

During the meeting, the client responded positively to Group 1's prototype, which was developed by Bruce and Zixiao Cheng. Their design was selected as the base project for future development. In addition, Group 1 had already implemented a gallery function, which aligned well with the client's expectations. As a result, Group 1 continued working on the gallery feature for the next iteration.The remaining team members work on different sections on the homepage.

Week 4

In Week 4, each team member was assigned responsibility for a different section of the homepage, all developed in line with the style established by Prototype 1. This division of work allowed us to progress in parallel while ensuring consistency across the homepage design. The sections included the hero area, prize section, outreach section, introduction, gallery, integration of components, and the event timeline.

Week 4 Task allocation

Ally Mui

- Took responsibility for client liaison by coordinating communication and organizing meetings.
- Worked on the hero section and prize section of the homepage to establish visual impact and highlight key achievements.

Bradley Zhang

- Focused on the outreach section, designing layout and content to present external engagement opportunities in a clear and accessible manner.

Bruce Wang

- Contributed to integration on the homepage, ensuring different components from sub-groups were combined smoothly.
- Also developed the gallery design, implementing functionality and structure for showcasing images.

Zixiao Cheng

- Worked on the introduction section, creating text and layout to provide visitors with an overview of the BIOTech Futures initiative.

Junhao Fu

- Developed TheBrief/AnnualReport section.

Kevin Lu

- Developed the new gallery design, refining layout and improving interactivity to better align with client requirements.

Qijun Zhou

- Took client meeting minutes and two team meeting minutes, ensuring clear documentation of discussions and decisions.
- Developed timeline section.

A link to the screenshots of each section developed by team members is provided here for reference:https://drive.google.com/drive/folders/1iQH4U7BT2IGfy4SOK6GYO6SnzDICLQwu?usp=sharing

Evidence on Client interactions via email week 4:
https://drive.google.com/drive/folders/1VJTvsxEmdMvgtmbXZGsIPLjE7Mi_tG29?usp=sharing

Client Feedback:

When we presented the completed homepage to the client, they were very satisfied with the outcome. The positive feedback confirmed that our design direction met their expectations, and they encouraged us to continue building additional pages of the website, such as events, mentors, and journey builder modules.

Week 5

Evidence on Client interactions via email week 5:
https://drive.google.com/drive/folders/1_-3DEwowZ0BALEeSRqkurEY3muRW8Ir8?usp=sharing

In Week 5, building on the client's positive feedback regarding our homepage design, we were asked to move forward to the design of subpages. Each team member was assigned responsibility for a different subpage to ensure balanced workload and parallel progress:

- Kevin Lu – News page
- Zixiao Cheng – Getting Started page
- Ally Mui – Past Winners and Sponsorship page
- Junhao Fu – Mentorship page and Documentation
- Bradley Zhang – Outreach page
- Qijun Zhou – Satellite pages (Victoria and Queensland)
- Bruce Wang – About and Contact Us pages

By dividing the subpages in this way, each member could focus on the design and functionality of their allocated section while maintaining consistency with the overall website style. This step marked the transition from the homepage prototype to the broader subpage development phase of the project.

The completed subpage designs were then compiled and presented to the client. A link containing screenshots of each member's subpage work has been provided here for reference:
https://drive.google.com/drive/folders/1xf30KknoYIUXvo_Reo1GBFX2dH0pQbFX?usp=sharing

Allocation of Roles

In alignment with XP practices, our group adopted a rotating role system, where roles such as tracker, manager, customer liaison, programmer, tester, and doomsayer were reassigned each week. This ensured that every member had the opportunity to experience different responsibilities, promoted knowledge sharing, and balanced the workload across technical and non-technical tasks.

Although the customer liaison role rotated internally, to reduce confusion for the client, Ally consistently served as the primary point of contact. This provided stability and minimized disruption in client communication. However, during weekly client meetings, both the current week's manager and customer liaison would actively participate in discussions with the client, ensuring that multiple team members were engaged in external communication while still keeping interactions efficient and consistent.

This structure allowed us to remain flexible internally while presenting a professional and reliable communication channel to the client.

2. Established Role

Week	Tracker	Manager	Customer Liaison	Programmer	Tester	Doomsayer
2	ZIYANG WANG	Chenxi Zhang	Ally Mui	WENWEI LU	Qijun Zhou	Zixiao Cheng Junhao Fu
3	Qijun Zhou	WENWEI LU Junhao Fu	Zixiao Cheng	Chenxi Zhang	ZIYANG WANG	Ally Mui
4	Zixiao Cheng	Junhao Fu	WENWEI LU Junhao Fu	Qijun Zhou	Ally Mui	ZIYANG WANG
5	Chenxi Zhang	Zixiao Cheng	ZIYANG WANG	Ally Mui	WENWEI LU	Qijun Zhou

Critique – Issues and Progress

From a teamwork perspective, our group has coordinated very effectively. Every member contributed actively, took ownership of their tasks, and worked with a positive attitude. In terms of group processes, our meetings were well-organized and highly communicative, with no major issues in collaboration. The use of XP practices such as continuous communication, collective ownership, and rapid feedback cycles helped us maintain steady progress and alignment across all members.

The main difficulty we faced at the beginning was not within the team, but with the client's unclear expectations. Their description of the desired style was often vague, using terms such as "something new" or "fresh" without providing specific design direction. This created initial uncertainty and made it challenging for us to decide on a consistent approach. However, by producing multiple prototypes and presenting concrete design options, we were able to guide the client toward a clearer vision and establish a solid foundation for development.

Overall, our progress demonstrates that while external ambiguity can cause early delays, strong teamwork, open communication, and adherence to XP principles enabled us to overcome these challenges and maintain momentum.

Team interactions:

- **Meeting Minutes**

  All client meetings, group contract, tutor meetings, and team meeting minutes have been recorded and are stored in our project repository. They can be accessed via the following GitHub link:

  https://github.sydney.edu.au/czha0443/SOFT3888_TH10_01_P012.git

  This ensured that all decisions, feedback, and task allocations were properly documented and accessible to every team member throughout the project.

- **Group Communication via Slack:**
  https://soft3888group01.slack.com/archives/C09B0N9926A

Client interaction via email and zoom:

- https://drive.google.com/drive/folders/1ymTtMW2B5HKOP6Tt5boFPrRx9CkJaYY7?usp=sharing
- https://us02web.zoom.us/launch/chat/v2/eyJzaWQiOiJ3ZWJfc2NoXzZkNzY1ZjVhY2U3ZDQ1OGE4MDlmNWNjMzBkZjY4ZjFlQGNvbmZlcmVuY2UueG1wcC56b29tLnVzIn0
- https://us02web.zoom.us/launch/chat/v2/eyJzaWQiOiJ3ZWJfc2NoXzZkNzY1ZjVhY2U3ZDQ1OGE4MDlmNWNjMzBkZjY4ZjFlQGNvbmZlcmVuY2UueG1wcC56b29tLnVzIn0

5 Feasible plan and risk mitigation strategy for the next project stage

5.1. Challenges / Risk Analysis

Main Challenges and Risks

- Incomplete feature: The prototype is completed, but several key features such as register and login, latest headlines and mentorship are still unstable or unfinished, which may affect demonstration and delivery quality.
- Compatibility: Differences across browsers, screen resolutions, and devices (especially mobile and iPad) may lead to inconsistent user experiences.
- Data and privacy compliance: The website may involve sensitive research data or user information, which is related to data protection regulations such as privacy policy, HTTPS, data minimization, etc.
- Performance risks: High-resolution images, potential bugs and heavy animation rendering may cause slow loading times, which may reduce user experience.
- Human resources and scheduling: Inappropriate time management and allocation of member responsibilities may cause project schedule delays or low efficiencies.
- Application deployment: Incorrect configuration of production environment and bad usage of deployment tools such as docker and Jenkins may cause deployment problems and delays.
- Scope creep: Client may add extra requirements which may push delivery dates.
- Integration with database and backend: Now the website has not connected to the database and has no backend and integration with database and backend in the later stage of development could have problems with compatibility, performance, scalability, etc.

Risk Level Assessment (Probability × Impact)

- High probability, high impact Compatibility issues, integration with databases and backend issues.
- Medium probability, medium impact: performance issues.
- Medium probability, high impact: Data and privacy compliance issues
- Low probability, high impact: Incomplete features.
- Low probability, medium impact: Human resources and scheduling, Application deployment issues.
- High probability, medium impact: Scope creep.

5.2. Limitations in terms of functionality, structure, design, implementation

Functionality

- No registration and log in functionalities are implemented.
- No authentication features or admin panels are implemented.

- Limited CRUD (create, read, update and delete) functionalities for entities such as mentors, past projects, user for data management
- Key functionalities such as contact, mentor, registration workflows are placeholders without databases and backend.

Structure

- No database and backend are used for data management which causes non-persistent data, bad data security, limited functionality, bad scalability.
- Some reusable components such as global styles are not designed as global components which cause code redundancy and bad maintainability.
- Embedded style tags are used in some pages which may cause poor maintainability of front-end styles.
- Different pages code files are put into the same folder and some pages are not divided into different sections.

Design

- No standardized and unified style such as spacing, typography scale, color and font size across different pages.
- Some page designs lack readability (bad text format) and usability (too many buttons) .

Implementation

- Most pages use placeholder data and don't support CRUD operations for data management.
- Heavy reliance on framer-motion and IntersectionObserver which may cause potential flicker/ on slow devices.
- Image resources have not been optimized (lack of responsive thumbnails) which causes performance issues.
- Tests are not yet covering end-to-end user flows (potential bugs) and no CI/CD tools implemented yet.

5.3. Technical and Other Constraints

Technical constraints

- Large image assets problem: Cause performance and bandwidth constraints.
- Browser compatibility: IntersectionObserver and some CSS features may behave differently across browsers (older Safari/iOS issues).
- Lack of authentication and data management backend: secure data handling and user management are constrained.
- Limited CI/CD: No CI/CD tools are used which may cause deployment and test problems.

Other constraints

- Time and human resource management: short remaining timeline for completing all the functionalities, final testing and deployment.
- Client availability: limited windows for review/feedback may delay development.

5.4. Group's Mitigation Strategy

Project management

- Contact with the clients as frequently as possible to get requirements and feedback on time.
- Make a clear schedule of the remaining works and use the Jira board for project management.

Technical methodology

- Implement responsive images (<img srcset> & sizes) and integrate an image optimizer (build-time plugin or CDN) to reduce asset sizes.
- Add progressive enhancement: ensure pages function without JS for critical content (contact info, event dates).
- Fallbacks for IntersectionObserver: add a polyfill or graceful degrade to avoid missing content reveals.
- Automated unit tests for key components, plus integration tests for routing.
- Implement CI pipeline: dev → unit tests → build → deploy to staging.

Security & privacy

- Add a backend with authentication and data management functionality to the system.
- If storing personal data, prepare a privacy policy and minimal data retention rules (comply with applicable laws).

5.5. List of user stories

Implemented: See the System Specification and Architecture Design part.

Not completed (with justifications and rationale)

1. Users can register and login to the website: Because there is no backend and database in the current system to store and manipulate user data.
2. User authentication: Because there is no backend and database in the current system to store and manipulate user data.
3. Users can check the latest headlines on the website: Time limits and this page needs lots of work.
4. Users can download and use the mobile app version of the website: Because the current web app system is not completed.

Discarded / merged (with justifications and rationale)

1. The user can have access to an introduction page with lots of text content in text boxed with interactive buttons: The clients thought there was too much information on one page, so this page is simplified and integrated into the home page.
2. The user can have access to a gallery page with an introduction to each picture: The clients don't like text in the gallery, so this page is modified to a more simplified gallery.

5.6. Key changes requested by the client

1. Design Style: The client didn’t have a style requirement of the new design, in email contact at the end of week 2, they gave us a style guide including unified colors, fonts, graphs and tables. After that, our designs followed the style guide to make sure styles are unified but there remained some ununified style problems.

2. Animation effect and photos: In week 3 and week 4 client meetings, the client requires us to add more animation effects and photos to the website so after that we added animation effects to some components on the website and added photos to text-only components.

3: Text organization: In the week 4 client meeting, the client thought there were too many texts on the website which caused bad readability so after that, we integrated some text into a dropdown text box and redesigned some text formatting to ensure readability.

5.7. Additional Work Beyond Scope & Minimum Requirements

- **React.js based web frontend:** The initial requirements ask us to use a non-coding tool Wix to improve the website but after discussion with client, the team rebuild the website with React.js which has better scalability, performance, maintainability, portability and has better integration compatibility with backend and database.
- **Animation effects and interactive UI:** The new website based on React.js has much more animation effects and interactive UI support by React libraries than the original Wix based website.
- **More functionality:** The new website based on React.js has more functionalities such as a gallery which supports grid, marquee and search modes, a more powerful FAQS page with search functionality, etc.
- **Better maintainability and scalability:** The new website based on React.js consists of each small tsx section component and can be easily integrated with backend, database and APIs compared with original Wix design.
- **Backend integration and mobile application:** The client also required us to integrate the frontend with backend and database as well as developing a mobile app version of the web app after the completion of frontend.

5.8. Feasible Plan & Schedule for Final Delivery

Objective: To deliver a fully functional, responsive, maintainable, and scalable BIOTech Futures website that meets client requirements, integrates with backend and database, and includes planned interactive features, animation effects, and mobile support, within the remaining project timeline .

Week 6 (Requirement Finalization & Planning)

- Confirm functional requirements with clients, including any last-minute adjustments to design style, animation, text organization, and gallery features.
- Finalize the detailed plan and role allocation for development, testing, and deployment using Jira boards.
- Prepare staging environment and CI/CD pipeline framework.

Week 7 (Core Frontend Development)

- Complete all remaining homepage and page sections in React.js, integrating animation effects, responsive layouts, and improved text formatting.
- Optimize images and static assets for performance.
- Begin frontend-backend API design for authentication, registration, and dynamic content.

Week 8 (Backend Development & Database Integration)

- Implement backend services including authentication, user management, mentorship, and content management APIs.
- Set up database schema for users, mentors, projects, and gallery items.
- Integrate frontend React components with backend APIs for data persistence and retrieval.

Week 9 (Mobile Support & Feature Completion)

- Develop mobile-friendly layouts and test responsiveness across devices (iOS, Android, tablets).
- Implement mobile app version of the website, ensuring functionality parity with the web version.

Week 10 (Testing & Quality Assurance)

- Conduct end-to-end testing of all user flows including registration, login, gallery interaction, and content updates.
- Perform cross-browser and cross-device compatibility testing.
- Test performance metrics: loading times, responsiveness, image optimization, and animation smoothness.
- Fix discovered bugs and performed regression testing.

Week 11 (Client Review & Feedback)

- Deploy staging version for client preview.
- Collect feedback on design, functionality, and readability.
- Conduct final testing and validation for all features.

Week 12 (Final Delivery & Deployment)

- Deploy the completed website to the production environment with proper CI/CD pipeline.
- Prepare documentation, tutorials, and developer guides for future maintenance.
- Deliver the product to the client.

6 Reflections and Conclusions

The project goals from our projects were threefold: usability, visual consistency and long-term sustainability. By adopting a modular React SPA with react-router-dom, framer-motion, and lucide-react, we improved navigation clarity, responsiveness, and component reuse. Iterative prototyping and client reviews aligned the design to BIOTech Futures' identity. The current system is maintainable and ready to extend, while keeping room for backend integration in future phases.

6.1 Critical Assessment of Development

We Worked well:

- **Prototype-first alignment**: Delivered multiple polished static UI prototypes early, letting the client choose style, type scale, color palette, and motion direction with minimal ambiguity.

- **Benchmark-informed design**: Compared peer sites to refine navigation, gallery flows, and content hierarchy, then adapted best patterns to our context.
- **Clear division of labour and parallel delivery**: Early stage: each member owned a homepage section; later stage: each owned a page, merging via GitHub for parallel progress.
- **Lightweight design system**: Established tokens/components/patterns (shared containers, section wrappers, color variables), increasing consistency and reuse.

We didn’t worked well:

- **Information density on the homepage**: Early versions packed too much text above the fold; visual hierarchy and chunking were insufficient, causing a “busy” first impression.
- **Typography & spacing drift**: Inconsistent type scale/line-height/margins across sections before tokens were enforced; micro-typography needed stricter rules.
- **accessibility**: Color contrast needed systematic checks.

6.2 Challenges / Risk analysis

- **Multi-platform (phone, tablet, desktop)**: Risk: layouts break or feel cramped on small screens. → Mitigation: responsive breakpoints, fluid grids, larger touch targets, test on iOS/Android & major browsers.

- **Accessibility & compatibility (motion, contrast, keyboard)**: Risk: some users can’t use/read the site. → Mitigation: prefers-reduced-motion, contrast checks (WCAG AA), visible focus, Tab-order tests.

- **Media weight & bandwidth**: Risk: slow loads and quota limits → Mitigation: compress images/video, lazy-load, use a CDN, and set image/video size budgets.

6.3 Primary Strengths

- Clear information architecture, mobile-first layouts, and consistent branding.
- Reusable components and tidy file structure enable fast iteration.
- Gallery built on a strategy-like core supports multiple modes and future data sources.
- Short feedback loops; decisions backed by prototypes and benchmarks.

6.4 Programming practices

- **Extreme Programming (XP).**
  Our team applied several XP principles to guide development. We broke down requirements into small user stories that could be completed within days, ensuring incremental progress and quick delivery of value. This allowed us to validate design choices step by step with the client. We followed a simple design philosophy, avoiding over-engineering and focusing only on what was needed for the next iteration. Code was subject to frequent refactoring, which kept the codebase maintainable as new features were added. Development proceeded in short iterations, each ending with a demo or review session.

  For higher-risk features such as routing logic, animations, and gallery components, we used pair programming and focused peer reviews to catch issues early. While test-driven development (TDD) was not consistently applied for UI, we maintained sanity checks and lightweight unit tests for build scripts and core logic. Ultimately, the greatest benefits of XP in our context came from continuous communication (both within the team and with the client) and rapid feedback cycles, which reduced misunderstandings and guided scope adjustments.

- **Version control & quality.**
  We adopted a GitHub Flow model: all work was done on feature/* branches, merged via pull requests (PRs), and reviewed before integration into main. We emphasized small, frequent PRs and used Conventional Commit messages for traceability. Issues were tracked with labels (feature, bug, design) and grouped into milestones (prototype, homepage, subpages) to visualize progress.

  For coding standards, we used ESLint and Prettier to enforce consistent style, which minimized unnecessary merge conflicts. Each PR was reviewed against a simple checklist covering accessibility, performance, and updated screenshots. A lightweight CI pipeline ran on every merge to ensure the build succeeded and that smoke tests for routing worked correctly. These practices helped the team maintain quality even as multiple members contributed in parallel.

- **Group & process.**
  We implemented rotating roles inspired by XP: tracker, manager, customer liaison, programmer, tester, and “doomsayer” (risk spotter). This rotation allowed everyone to experience different responsibilities and share knowledge. However, to avoid confusing the client, Ally consistently served as the stable external contact, while the week’s manager joined her in client meetings.

  Within the team, we held weekly stand-ups and show-and-tell demos to synchronize progress and discuss challenges. All important discussions were documented in meeting minutes, and risks were tracked openly in our shared repository. This combination of structured communication and flexible role distribution helped us stay aligned, balance workload, and maintain accountability throughout the project.

6.5 Further Improvement

- **Testing**: Add smoke and end-to-end tests for routing, gallery filter/lightbox, and form flows.

- **Content operations**: Move text and assets into a simple CMS or config files for easier updates.

- **Deployment**: Use stable hosting/CDN, set up basic CI, and add cache headers.

- **Documentation**: Improve the README, provide one-command setup, .nvmrc and lockfiles, and include style guide and tokens.

- **Animations**: Add more polished and engaging animations while keeping good performance and accessibility.

7 Individual contributions (individual reports), work split

Table of contributions

Name	Contribution Evidence							
ZIYANG WANG	Improved the About subpage with "meet the team" · czha0443/SOFT3888_TH10_01_P012@bb276f7	Complete the basic version for the About page · czha0443/SOFT3888_TH10_01_P012@0480e60	Updated the gallery functionality for better sorting and searching · czha0443/SOFT3888_TH10_01_P012@3cff5b9	First integration on navigation bar with animation · czha0443/SOFT3888_TH10_01_P012@b444756	Initial Commit: BIOTech Future Gallery prototype · czha0443/SOFT3888_TH10_01_P012@e1cceba	ZIYANG's Subpages - Google Drive	SOFT3888_TH10_01_P012 - Google Docs	portotype1 - Google 云端硬盘
Hiolam Mui	https://github.sydney.edu.au/czha0443/SOFT3888_TH10_01_P012/commits/Documentation/Week2_client_meeting.pdf	czha0443/SOFT3888_TH10_01_P012@154e3cd — Week 5 Team minutes.pdf	Commited_new_past_winner	uploaded_sponsorpage	Pr…	…	A…	Google_doc_history_of_contribution
Qijun Zhou	Add timeline · czha0443/SOFT3888_TH10_01_P012@abe494d	Queensland satellite completed except the video and contract · czha0443/SOFT3888_TH10_01_P012@156aa86	Queensland satellite done · czha0443/SOFT3888_TH10_01_P012@1a5380a	Victoria satellite done · czha0443/SOFT3888_TH10_01_P01	Merge pull request #32 from czha0443/QIJUN · czha0443/SOFT3888_TH10_01	Add files via upload · czha0443/SOFT3888_TH10_01_P012@b76436f	Add files via upload · czha0443/SOFT3888_TH10_01_P012@3886afc	Qijun's subpage - Google Drive

				2@251477c	_P012@bc3e038			
Junhao Fu	update Mentorship subpage	update TheBriefSection	Upload week5 team meeting2 minutes	Upload week 5 client meeting minutes	update AnnualReportSection	P…		
Chenxi Zhang	Merge pull request #41 from czha0443/ChenxiZhang · czha0443/SOFT3888_TH10_01_P012@f4badfd	Merge pull request #31 from czha0443/ChenxiZhang · czha0443/SOFT3888_TH10_01_P012@d6f29e7	update outreach and refine education · czha0443/SOFT3888_TH10_01_P012@bf0646a	Merge pull request #26 from czha0443/ChenxiZhang · czha0443/SOFT3888_TH10_01_P012@bf4de14	Bradley's Section - Google 云端硬盘	Merge pull request #16 from czha0443/ChenxiZhang · czha0443/SOFT3888_TH10_01_P012@dc87b2a	Bradley's Subpages - Google 云端硬盘	Prototype2 - Google 云端硬盘
Zixiao Cheng	Added a navigation bar above the gallery page and completed the inter… · czha0443/SOFT3888_TH10_01_P012@afcd29d	Add headers and footers to the gallery section · czha0443/SOFT3888_TH10_01_P012@13b2c05	improve intro function · czha0443/SOFT3888_TH10_01_P012@cf2be63	add whatis section · czha0443/SOFT3888_TH10_01_P012@6806a23	Add Getting Start page and FAQpage to the website · czha0443/SOFT3888_TH10_01_P012@0828129	Zixiao's Section - Google 云端硬盘	Zixiao’s subpage - Google 云端硬盘	portotype1 - Google 云端硬盘
WENWEI LU	fixed bug of unseen scroll bar	Integrate gallery.html and	prototype4	https://drive.goo	https://drive.google.com/d			

	https://github.sydney.edu.au/czha0443/SOFT3888_TH10_01_P012/commit/2c3c21adf22de43ef92f728021bd431510b7d7ca	navigation updates https://github.sydney.edu.au/czha0443/SOFT3888_TH10_01_P012/commit/70afddbde00e8d5c268116291f320c47d00ed092	https://github.sydney.edu.au/czha0443/SOFT3888_TH10_01_P012/commit/97aa583d3f929fa9243902b662046f6defce0433	gle.com/drive/folders/1DynuRsDBaeGvQZpDSYissN6Tfno1wwgY?usp=drive_link	rive/folders/1ASRBd6IcMUGdUKewbfvYTauKvKhkPp8j?usp=drive_link			

Appendix (Page limit – 10 pages)

Appendix A — Client Interactions

- **Week 2 Emails**: Google Drive Folder
- **Week 2 Client Meeting Zoom Chat**: Zoom Chat Link
- **Week 3 Emails**: Google Drive Folder
- **Week 3 Client Meeting Zoom Chat**: Zoom Chat Link
- **Week 4 Emails**: Google Drive Folder
  Week 5 Emails: Google Drive Folder
- **General Client Communication (emails & Zoom)**:
  - Google Drive Folder
  - Zoom Chat Link 1
  - Zoom Chat Link 2

Appendix B — Task Allocation & Documentation

- **Week 2 Task Allocation Evidence**: Google Doc
- **Week 3 Prototypes & Gallery Design**: Google Drive Folder
- **Week 4 Section Screenshots**: Google Drive Folder
- **Week 5 Subpage Screenshots**: Google Drive Folder

Appendix C — Team Interactions

- **Meeting Minutes & Project Repository**: GitHub Repository
- **Slack Communication**: Slack Workspace

Details to run the code:

Github: SOFT3888_TH10_01_P012/Prototype1 at main · czha0443/SOFT3888_TH10_01_P012
Procedure:

1. **Clone the repository**
   Open a terminal and run:git clone
   https://github.sydney.edu.au/czha0443/SOFT3888_TH10_01_P012.git

   cd SOFT3888_TH10_01_P012/Prototype1

2. **Install dependencies**

   Ensure you have **Node.js (v16 or later)** and **npm (or yarn)** installed.

   Then install the required packages: npm install

3. **Start the development server**

   Launch the local server with: npm run dev

Note: *If you encounter errors, make sure your Node.js version matches the project's requirements, and delete/reinstall node_modules if necessary.*

References:

Reference for graphic design

- ***Anything - AI app builder*. (n.d.). Anything - AI App Builder. https://www.createanything.com/**
- ***Unseen Studio® – Brand, Digital & Motion*. (2022, November 22). Unseen Studio®.**

  https://unseen.co/