A black and gold logo

AI-generated content may be incorrect.

INFO8003

Final Major Project (Afterschool Ally)

Developing a Business System (Mobile Application)

Submitted by: -

Group 4

Table of Contents

[Using a collaborative tool: GitHub Collaboration Plan - AfterSchool Ally 2](#_Toc204810305)

[Repository Setup 2](#_Toc204810306)

[File Structure: 2](#_Toc204810307)

[README.md 4](#_Toc204810308)

[Team Roles 5](#_Toc204810309)

[Specific Tasks 5](#_Toc204810310)

[Forming a Development Team 6](#_Toc204810311)

[Team Role Allocation 6](#_Toc204810312)

[Scrum Roles 6](#_Toc204810313)

[Task Allocation 7](#_Toc204810314)

[Preliminary Research 8](#_Toc204810315)

[Research Objectives 8](#_Toc204810316)

[Research Methods 8](#_Toc204810317)

[Findings 9](#_Toc204810318)

[Conclusion 10](#_Toc204810319)

[Brainstorming 11](#_Toc204810320)

[Benefits of the Application 11](#_Toc204810321)

[Feature Ideas 11](#_Toc204810322)

[How the Application Works 13](#_Toc204810323)

[Summary 13](#_Toc204810324)

[Research – Web Market Analysis 14](#_Toc204810325)

[Investigation of Existing Applications 14](#_Toc204810326)

[Similar Market Applications 17](#_Toc204810327)

[Features Needed for AfterSchool Ally 18](#_Toc204810328)

[Differences from Existing Applications 18](#_Toc204810329)

[Rationale for Development 18](#_Toc204810330)

[Description (10 marks) 20](#_Toc204810331)

[Refined Description 20](#_Toc204810332)

[Technical Requirements 23](#_Toc204810333)

[Special Device Hardware Features 23](#_Toc204810334)

[Discussion: 23](#_Toc204810335)

[Special Input Controls 23](#_Toc204810336)

[Discussion: 23](#_Toc204810337)

[Contextual Constraints and Advantages 24](#_Toc204810338)

[Constraints: 24](#_Toc204810339)

[Advantages: 24](#_Toc204810340)

[Personas 25](#_Toc204810341)

[Persona 1: Parent - Maria Gonzalez 25](#_Toc204810342)

[Persona 2: Volunteer - Li Wei 25](#_Toc204810343)

[Persona 3: Admin - Sarah Thompson 26](#_Toc204810344)

[Use Cases 27](#_Toc204810345)

[Use Case List 27](#_Toc204810346)

[Detailed Use Case Descriptions 28](#_Toc204810347)

[Use Case 1: Earn Points 28](#_Toc204810348)

[Use Case 2: View Feedback 28](#_Toc204810349)

[Activity Diagram 30](#_Toc204810350)

[Diagram Screenshot: 30](#_Toc204810351)

[Entities 32](#_Toc204810352)

[ERD Diagram: 32](#_Toc204810353)

[Content Plans 34](#_Toc204810354)

[Page 1: Earn Points 34](#_Toc204810355)

[Page 2: View Feedback 34](#_Toc204810356)

[Prototypes 36](#_Toc204810357)

[Screenshot below 36](#_Toc204810358)

[Class Diagram 37](#_Toc204810359)

[Screenshot below 37](#_Toc204810360)

[Implementation - Developing the Agile Plan 38](#_Toc204810361)

# Using a collaborative tool: GitHub Collaboration Plan - AfterSchool Ally

The team is to use a tool such as **GitHub**, Azure or AWS to collaboratively develop and record your materials.

At the end of the project, your team will upload a **document** containing all of the deliverables, but you will also include in the document **a link to your repository** which your instructor will be able to use to view all documents.

## Repository Setup

* **Repository**: <https://github.com/pengzishang/MajorProject>
* **Branch**: Use only the main branch.

## ****File Structure****:

MajorProject/

├── docs/

│ └── final\_report.docx # Word document consolidating all deliverables

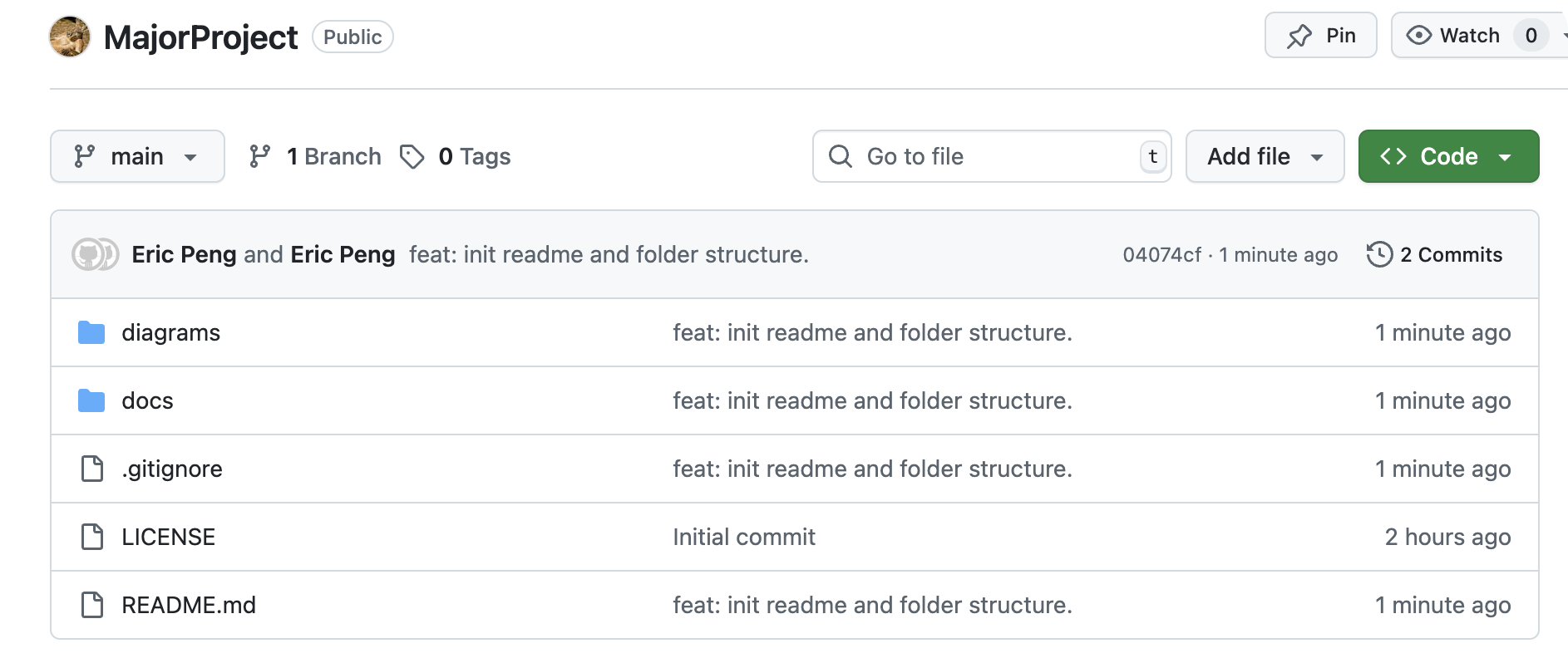
├── diagrams/

│ ├── erd\_lucidchart.pdf # Entity Relationship Diagram (exported from LucidChart)

│ └── class\_diagram\_lucidchart.pdf # Class Diagram (exported from LucidChart)

├── README.md # Repository description

└── .gitignore # Ignore unnecessary files



## README.md

A screenshot of a computer

AI-generated content may be incorrect.

## Team Roles

* **Zhuoran Zhang (Product Owner)**:
  + Coordinate project progress and define platform vision.
  + Contribute the introduction, conclusion, and personas to the Word document.
  + Commit message example: Add introduction and personas to final\_report.docx by Zhuoran.
* **Zishang Peng (Scrum Master)**:
  + Set up repository (create README.md and .gitignore).
  + Design and export ERD and Class Diagram (LucidChart).
  + Contribute technical requirements and agile plan to the Word document.
  + Commit message example: Add ERD diagram (erd\_lucidchart.pdf) by Zishang.
* **Shubham Verma**:
  + Write the main content of the Word document (research, market analysis, use cases, prototype description).
  + Proofread the document for professional English.
  + Commit message example: Add market analysis and use cases to final\_report.docx by Shubham.

## Specific Tasks

1. **Word Document (/docs/final\_report.docx)**:
   * Consolidate all deliverables: introduction, research, market analysis, personas, use cases, content plan, agile plan, and conclusion.
   * Include descriptions referencing LucidChart diagrams.
   * Shubham leads writing, Zhuoran contributes the introduction and personas, and Zishang provides technical sections.
2. **ERD Diagram (/diagrams/erd\_lucidchart.pdf)**:
   * Zishang designs in LucidChart, exports as PDF.
   * Example entities: Child, Volunteer, Activity, Donation, Grant.
3. **Class Diagram (/diagrams/class\_diagram\_lucidchart.pdf)**:
   * Zishang designs in LucidChart, includes at least 5 classes (Child, Volunteer, Activity, Donation, Grant), and exports as PDF.

# Forming a Development Team

Imagine that your team is a small start-up company and you are developing the system for which you have developed a concept. Your team is now about to work through all the tasks that have been identified in Projects 1 through 4.

Identify roles for your team.

## Team Role Allocation

Based on team members' backgrounds (Zhuoran: leadership, new to software; Zishang: experienced developer; Shubham: native English, new to software), the following roles and tasks are assigned:

### Scrum Roles

* **Product Owner**: Zhuoran Zhang
  + **Responsibilities**: Define AfterSchool Ally vision, ensure features meet nonprofit needs, coordinate team, and communicate with instructors.
  + **Rationale**: Zhuoran's leadership skills suit project direction; English support from Shubham aids documentation and presentation.
* **Scrum Master**: Zishang Peng
  + **Responsibilities**: Manage agile process, oversee sprint planning, ensure timely task completion, and review technical documents/diagrams.
  + **Rationale**: Zishang’s development experience and strong English skills suit technical tasks and process management.
* **Development Team**: Zhuoran Zhang, Zishang Peng, Shubham Verma
  + All members contribute to tasks, with specific roles outlined below.

### Task Allocation

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Primary** | **Support** | **Description** |
| Brainstorming | Zhuoran | Shubham, Zishang | Zhuoran leads idea generation (e.g., gamified points), Shubham records, Zishang assesses technical feasibility. |
| Research | Shubham | Zhuoran | Shubham researches education nonprofits and competitors, Zhuoran provides community insights. |
| Business Model | Zhuoran | Shubham | Zhuoran defines vision and revenue streams, Shubham writes description. |
| Market Analysis & Personas | Shubham | Zhuoran | Shubham writes analysis and personas, Zhuoran adds low-income family perspectives. |
| Use Cases | Shubham | Zishang | Shubham drafts use case descriptions, Zishang reviews technical details. |
| ERD | Zishang | Shubham | Zishang designs ERD (LucidChart), Shubham documents entity attributes. |
| Class Diagram | Zishang | Shubham | Zishang designs class diagram (LucidChart), Shubham documents methods/attributes. |
| Content Plans | Shubham | Zhuoran | Shubham plans prototype page content, Zhuoran reviews user experience. |
| Prototypes | Shubham | Zhuoran | Shubham designs prototypes in Figma, Zhuoran reviews usability. |
| Code Development | Zishang | Shubham | Zishang plans code structure, Shubham documents pseudocode/process. |
| Testing | Zishang | Shubham | Zishang designs testing strategy, Shubham documents test cases. |
| Documentation | Shubham | Zhuoran, Zishang | Shubham compiles Word document, Zhuoran contributes intro/conclusion, Zishang reviews technical sections. |

# Preliminary Research

You may use AI tools to help you with this to investigate topics of interest using the list provided above as a guide (though you are certainly not limited to this). ***In an Appendix***, please **provide the prompts** you used for this. **Make sure you understand the results from AI**. ***Correct it and adapt it to your purposes***. ***Make sure that you do not make any incorrect claims*** about work that has been done or will be done.

## Research Objectives

Gather background information for AfterSchool Ally (a nonprofit platform for afterschool education for low-income children), focusing on:

* Operational models of education nonprofits.
* Features and gaps in afterschool program platforms.
* Gamified education and incentive mechanisms.
* Educational needs and challenges in low-income communities.

## Research Methods

* **Secondary Research**: Analyzed public reports, websites, and statistics on education nonprofits and afterschool programs. Sources include:
  + National Center for Charitable Statistics (NCCS): Data on nonprofit size and donation trends.
  + Nonprofit Impact Matters: Insights into nonprofit operations.
  + Research for Action: Studies on out-of-school time (OST) program effectiveness.
  + Competitor Analysis: Reviewed platforms like Boys & Girls Club, KinderCare, and YMCA for event management, volunteer coordination, and donor tracking features.
* **AI-Assisted Research**: Used AI tools (e.g., Grok) to query afterschool program management tools and gamified education cases, validating and adapting results for project needs (see Appendix).
* **Team Brainstorming**: Discussed findings to identify AfterSchool Ally’s unique features, such as multilingual support, parent communication, and a gamified points system.

## Findings

* **Nonprofit Overview**:
  + Education nonprofits (e.g., Boys & Girls Club) rely on foundations, corporate sponsors, and government grants, with annual budgets ranging from tens of thousands to millions of dollars (NCCS, 2023).
  + Volunteers are critical, requiring efficient coordination for activities and teaching.
* **Competitor Analysis**:
  + Boys & Girls Club offers event management and volunteer coordination, but lacks real-time parent communication.
  + KinderCare’s platform is feature-rich but commercialized and unaffordable for low-income families.
  + YMCA provides basic event management but lacks gamification and impact reporting.
  + Gaps: Existing platforms lack multilingual support and free resources for low-income families.
* **Gamification Mechanisms**:
  + Platforms like Classcraft use points and rewards to boost student engagement, increasing attendance by 20% (Research for Action, 2022).
  + AfterSchool Ally can implement a points system where children earn points for completing tasks or activities, redeemable for stationery or priority registration.
* **Community Needs**:
  + Low-income families face high costs and language barriers in accessing afterschool programs (Nonprofit Impact Matters, 2024).
  + Needs include flexible schedules, free resources, and multilingual interfaces for diverse communities (e.g., Spanish, Chinese users).

## Conclusion

The research highlights that AfterSchool Ally can address gaps in existing platforms through multilingual support, parent communication, and gamified incentives, meeting the afterschool education needs of low-income families. The next phase will design features and prototypes based on these findings.

# Brainstorming

Develop and document some ideas for the ***benefits*** ***of your application concept*** and how the application might work. Provide a ***summary*** of your ideas - these can be changed later. Al is permitted with the usual caveats.

## Benefits of the Application

AfterSchool Ally aims to provide free afterschool education for low-income children, addressing educational inequality. Key benefits include:

* **Enhanced Educational Equity**: Offers free academic tutoring and enrichment activities, reducing the education gap.
* **Increased Community Engagement**: Encourages community support through volunteer coordination and parent communication.
* **Motivated Child Participation**: Gamified points system boosts attendance and learning enthusiasm.
* **Attracting Funding**: Visualized impact reports showcase outcomes, increasing foundation and corporate donations.
* **Multilingual Support**: Provides English, Spanish, and Chinese interfaces, serving diverse communities and reducing language barriers.

## Feature Ideas

Through team brainstorming and AI assistance (prompts below), the following features were proposed:

* **Event Management Module**:
  + Admins create afterschool activities (e.g., math tutoring, art workshops), setting time, location, and registration requirements.
  + Parents register via the platform, receiving SMS/email reminders.
  + Idea: Activity calendar with filters (e.g., by age or interest) for easy parent selection.
* **Volunteer Coordination System**:
  + Volunteers register, providing skills (e.g., teaching, sports) and availability.
  + System matches volunteers to activities based on skills and location.
  + Idea: Volunteers earn “contribution badges” to encourage long-term engagement.
* **Gamified Incentive Mechanism**:
  + Children earn points for completing homework or activities, redeemable for rewards (e.g., stationery, priority registration).
  + Idea: Points leaderboard for children to view rankings, adding competitive fun.
* **Parent Communication Module**:
  + Parents view children’s attendance, homework completion, and teacher feedback.
  + Idea: Real-time notifications (e.g., “Your child completed math homework”) to enhance interaction.
* **Donor Tracking and Impact Reporting**:
  + Tracks donations (cash, supplies), generating personalized thank-you letters.
  + Auto-generates visualized reports (e.g., attendance, progress charts) for grant applications.
  + Idea: Donor-specific page showing how donations help individual children.

## How the Application Works

AfterSchool Ally is a cloud-based platform accessible via web and mobile:

**User Roles**: Admins (nonprofit), volunteers, parents, children, donors.

* **Workflow**:
  + Admins create activities; volunteers register and are matched.
  + Parents enroll children in activities and track progress.
  + Children participate, earn points, and redeem rewards.
  + Donors contribute and view impact reports; admins generate reports for grants.
* **Technical Implementation**:
  + Frontend: React + Tailwind CSS for responsive interfaces.
  + Backend: Node.js + MongoDB for user and activity data.
  + AI Integration: Recommends activities and volunteer matches, generates report visuals.
* **AI-Assisted Brainstorming**: Used AI tools (e.g., Grok) to generate feature ideas, validated and adapted by the team for relevance. For example, the AI-suggested “points system” was adjusted to offer low-cost rewards (e.g., stationery instead of expensive prizes).

## Summary

AfterSchool Ally addresses low-income children’s educational needs through event management, volunteer coordination, gamified incentives, parent communication, and impact reporting. Features emphasize user-friendliness and community engagement, with technical implementation focusing on scalability. The next phase will refine features and develop prototypes based on these ideas.

# Research – Web Market Analysis

Research should be done to investigate the field of application. Do such apps already exist? Provide links and screenshots wherever possible. If it does not exist, what applications would be of interest to a similar market? What might people use now to accomplish the same goals, if anything? Describe the features needed by your app and explain how yours would be different from existing apps. Why does it make sense to develop this app?

## Investigation of Existing Applications

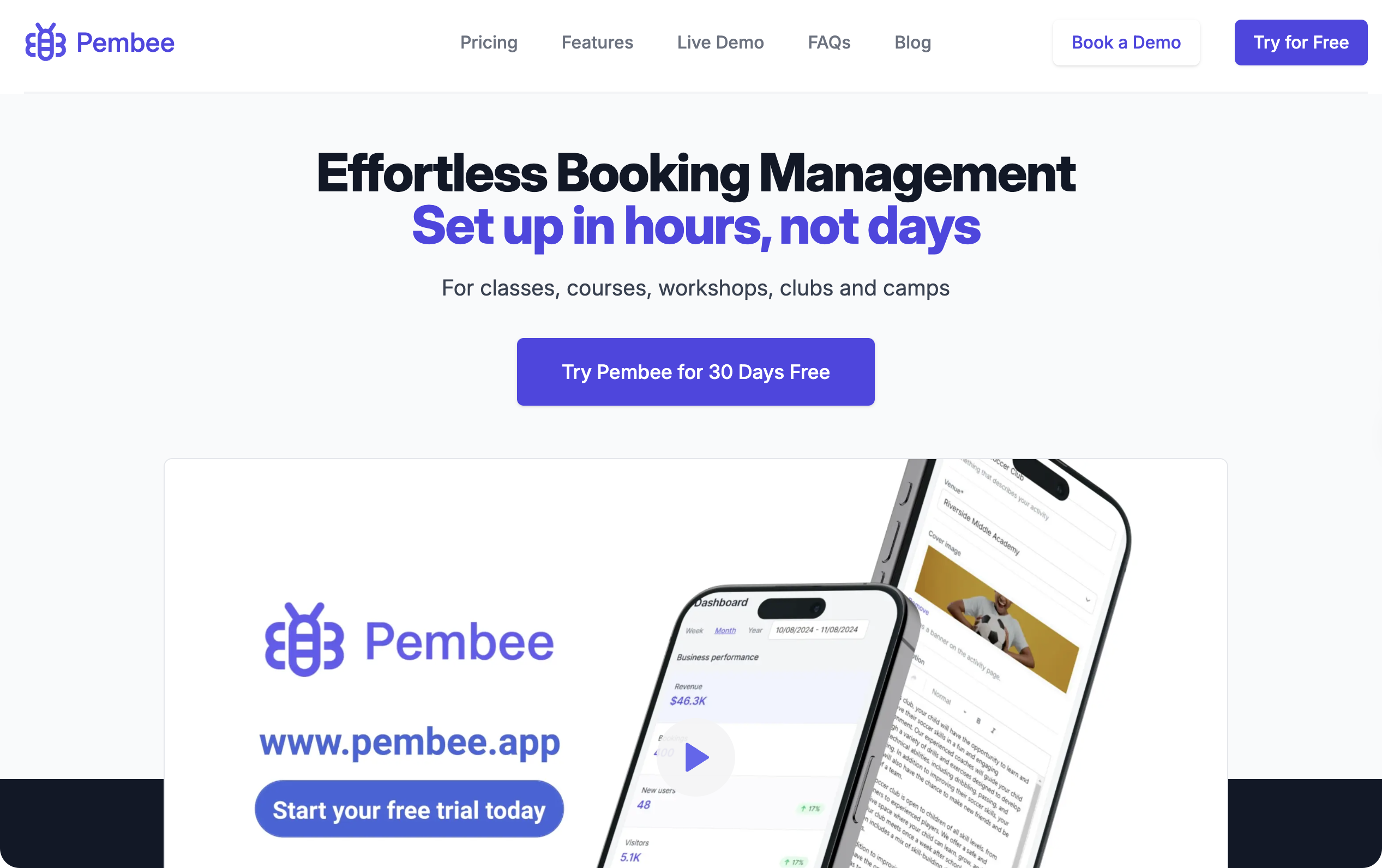
Through web research, we investigated existing afterschool program management applications, focusing on platforms serving nonprofits and low-income families. Key findings (as of July 29, 2025):

1. **AfterSchool HQ** ([https://go.afterschoolhq.com](https://go.afterschoolhq.com/))



* + **Features**: Offers online registration, payment processing, attendance tracking, and parent communication. Supports custom forms, QR code check-ins, and data reports for grant applications.
  + **Applicability**: Ideal for nonprofits and schools, streamlining administrative tasks.
  + **Limitations**: Lacks gamified incentives, has no clear multilingual support, and pricing may be prohibitive for low-income families.

1. **Pembee** ([https://www.pembee.app](https://www.pembee.app/))



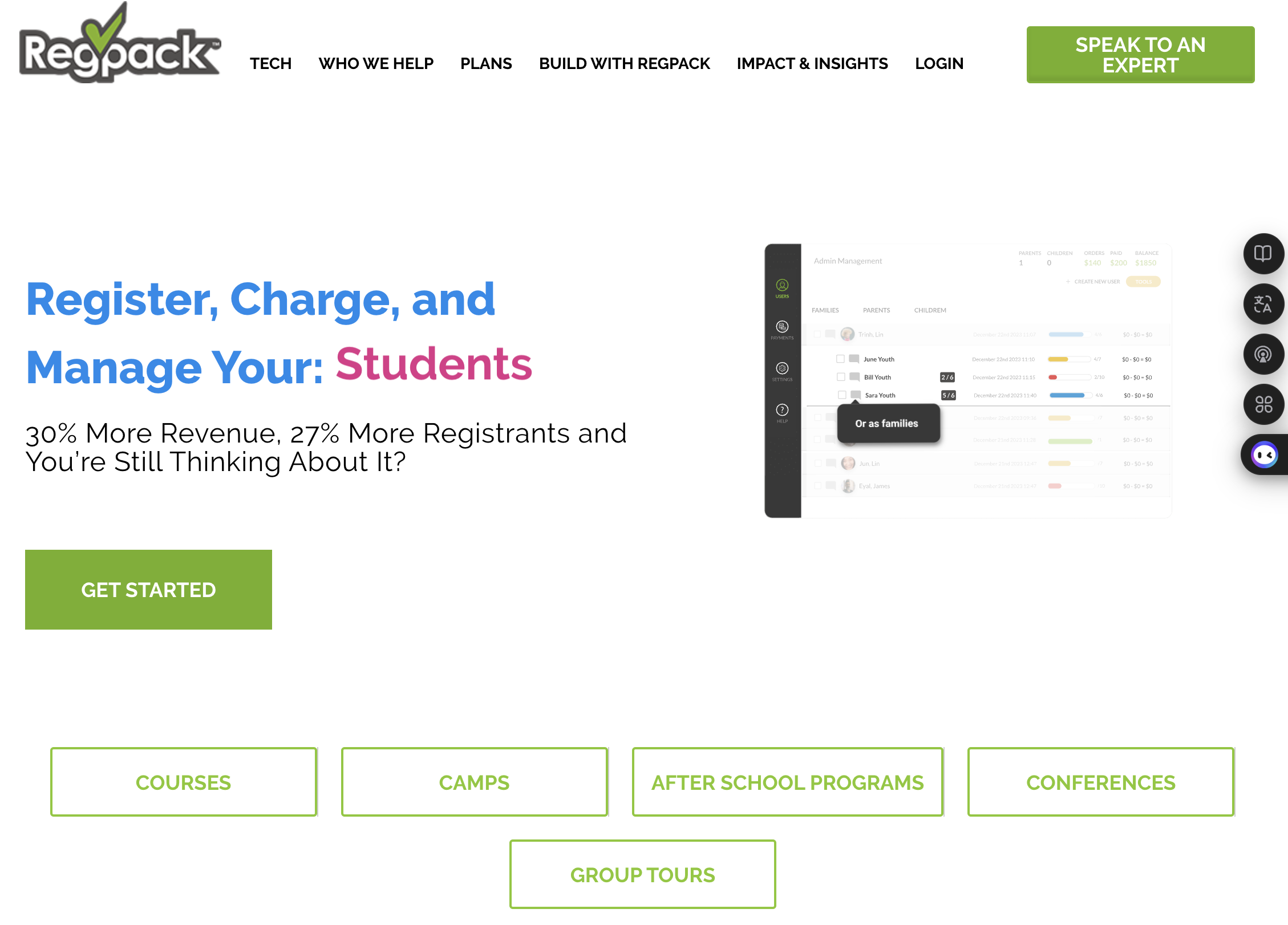
* + **Features**: Provides customizable registration forms, flexible payments (e.g., installments), real-time attendance, and discount rules. Parents can update child details (e.g., allergies).
  + **Applicability**: Enhances parent experience, reduces admin workload.
  + **Limitations**: No gamification, lacks free resources for low-income families, and has limited language options.

1. **EZChildTrack** ([https://www.ezchildtrack.com](https://www.ezchildtrack.com/))



* + **Features**: Supports online registration, billing, parent portal, and electronic check-in/out. Includes calendar scheduling, auto-billing, and absence tracking.
  + **Applicability**: Suits small to medium schools and programs, emphasizing safety and data management.
  + **Limitations**: Focuses on admin tasks, lacks incentives and multilingual support, and pricing may be high ($500/year/school).

1. **Regpack** ([https://www.regpacks.com](https://www.regpacks.com/))



* + **Features**: Offers conditional logic forms, automated emails, discount management, and data analytics. Supports mobile registration and parent data updates.
  + **Applicability**: Suitable for diverse afterschool programs, improving registration efficiency.
  + **Limitations**: Commercial focus, no free options or gamification, not optimized for nonprofits.

## Similar Market Applications

For low-income families, no dedicated afterschool program apps may exist, so they likely use:

1. **General Communication Tools**: WhatsApp or Google Calendar for activity coordination.
   * **Limitations**: Lack professional management features (e.g., attendance, donation tracking), inefficient.
2. **School Management Systems**: PowerSchool or ClassDojo for attendance and communication.
   * **Limitations**: Designed for schools, missing nonprofit-specific features like volunteer coordination or impact reporting.
3. **Community Bulletin Boards**: Nextdoor or Facebook groups for activity announcements.
   * **Limitations**: Scattered information, no registration or data management.

## Features Needed for AfterSchool Ally

Based on market analysis, AfterSchool Ally requires:

1. **Event Management**: Admins create activities (tutoring, enrichment), parents register via filters (e.g., age, interest) and receive notifications.
2. **Volunteer Coordination**: Volunteers register skills and availability, the system matches them to activities, and offers “contribution badges” for engagement.
3. **Gamified Incentives**: Children earn points for activities, redeemable for stationery or priority registration, with leaderboards for fun.
4. **Parent Communication**: Parents view attendance, homework, and feedback, receive real-time notifications, with multilingual support (English, Spanish, Chinese).
5. **Donor Tracking and Impact Reporting**: Tracks donations, generates visualized reports (e.g., attendance charts) to attract funding.

## Differences from Existing Applications

AfterSchool Ally differs from existing apps by:

1. **Free and Nonprofit-Oriented**: Unlike paid apps (e.g., Pembee, EZChildTrack), AfterSchool Ally is free, designed for low-income families.
2. **Gamified Incentives**: Existing apps lack child engagement mechanisms; AfterSchool Ally introduces points and leaderboards to boost participation (inspired by Classcraft, Research for Action, 2022).
3. **Multilingual Support**: Unlike apps with limited language options, AfterSchool Ally offers English, Spanish, and Chinese interfaces for diverse communities.
4. **Impact Reporting**: AfterSchool Ally provides donor-specific pages and automated reports, enhancing funding appeal beyond AfterSchool HQ’s basic reporting.

## Rationale for Development

Developing AfterSchool Ally is justified because:

1. **Market Need**: Demand for free after-school programs for low-income families is unmet, as existing apps are costly and not tailored (Afterschool Alliance, 2024).
2. **Educational Equity**: Providing free education support reduces gaps, aligning with social values (Youth.gov, 2025).
3. **Funding Potential**: Impact reports attract grants, as seen with Felege Hiywot Center securing $2M using AfterSchool HQ (AfterSchool HQ, 2024).
4. **Technical Feasibility**: Built on scalable technologies (React, Node.js, MongoDB), with AI recommendations enhancing efficiency.

# Description (10 marks)

Provide a **description of the application**, and **its various purposes** – both **business and social.** Delineate the ideas that you are bringing to the application. Also, explain **future goals** for the app – in other words, if you were to develop the idea beyond this course, **what new features, extensions or purposes would you add to it**? (For example, could the app be adapted for a similar but different purpose?) Explain **how it relates to the theme**. Provide:

* a **refined** description
* a **preliminary list of desirable features** for the app (might change)
* possible **revenue**
* future **goals**

## Refined Description

AfterSchool Ally is a free cloud-based platform designed for nonprofits to support afterschool education for low-income children. It enhances educational equity, encourages child participation, and fosters community and donor support through event management, volunteer coordination, gamified incentives, parent communication, and donor tracking. The platform offers multilingual interfaces (English, Spanish, Chinese) to ensure inclusivity for diverse communities.

#### Business and Social Purposes

* **Business Purposes**:
  + Provide nonprofits with efficient tools to streamline event management, volunteer allocation, and donation tracking, reducing administrative costs.
  + Attract foundation and corporate funding through visualized impact reports, ensuring financial sustainability.
* **Social Purposes**:
  + Offer free afterschool education to low-income children, reducing educational disparities (Afterschool Alliance, 2024).
  + Boost child attendance and motivation through gamified incentives (inspired by Classcraft, Research for Action, 2022).
  + Foster community engagement by connecting parents, volunteers, and donors in a supportive network.

#### Preliminary Feature List

Based on brainstorming and market analysis, AfterSchool Ally includes the following features (subject to change):

1. **Event Management**: Admins create tutoring or enrichment activities; parents register via filters (age, interest) and receive SMS/email notifications.
2. **Volunteer Coordination**: Volunteers register skills and availability; system matches them to activities and awards “contribution badges” for engagement.
3. **Gamified Incentives**: Children earn points for completing activities, redeemable for stationery or priority registration, with leaderboards for fun.
4. **Parent Communication**: Parents view attendance, homework, and feedback, receive real-time notifications, with multilingual support.
5. **Donor Tracking and Impact Reporting**: Tracks donations, generates visualized reports (e.g., attendance charts), and offers donor-specific pages to showcase contributions.

#### Possible Revenue Sources

* **Foundation Service Fees**: Charge low management fees (e.g., 2-5% of donations) to funding foundations for platform maintenance (similar to AfterSchool HQ).
* **Corporate Sponsorships**: Partner with businesses to sponsor activities or rewards (e.g., stationery donations) in exchange for brand exposure.
* **Community Crowdfunding**: Launch small crowdfunding campaigns on the platform to support specific projects.
* **Government Grants**: Apply for education or community development grants, leveraging impact reports to demonstrate outcomes.

#### Future Goals

If developed beyond this course, AfterSchool Ally would expand with:

1. **Online Course Module**: Add virtual tutoring and enrichment courses (e.g., coding, arts) for remote access, broadening reach.
2. **Youth Career Training**: Introduce career skills courses (e.g., resume writing, basic coding) to prepare teens for employment.
3. **Cross-Region Expansion**: Partner with more nonprofits to scale the platform to other cities or countries.
4. **AI-Enhanced Features**: Use AI for optimized volunteer matching (e.g., skill-based algorithms) and personalized learning plans.
5. **Adapted Purposes**: Repurpose the platform for other nonprofit sectors, such as senior community activities or disability support, retaining event and donation tracking features.

#### Relation to Theme

AfterSchool Ally directly aligns with the “nonprofit afterschool education support” theme, addressing low-income children’s needs with free educational resources, promoting educational equity. Its gamified incentives and multilingual support fill gaps in existing platforms (e.g., AfterSchool HQ, Pembee), enhancing community engagement and funding potential.

# Technical Requirements

* What special device hardware features would the app require, if any? (If none, please provide **discussion**.)
* What special input controls would you develop for this application, if any? (If none, please provide **discussion**.)
* What **constraints or advantages** are involved in this app with context?

## Special Device Hardware Features

AfterSchool Ally is a cloud-based platform accessible via web and mobile, requiring no special device hardware features.

### **Discussion**:

* **Hardware Needs**: The platform is designed to work on common devices (e.g., smartphones, tablets, laptops) without requiring specialized hardware (e.g., high-performance GPUs or sensors). Users (parents, volunteers, admins) may use low to mid-range devices (e.g., Android 6.0+ or iPhone 6s+), so the platform uses a lightweight frontend framework (e.g., React) for performance.
* **Rationale**: Low-income families may lack access to high-end devices, so compatibility with a wide range of devices ensures accessibility (Afterschool Alliance, 2024, data on low-income device usage).
* **Optional Feature**: Future support for QR code scanning (for event check-ins) would use standard cameras, available on most smartphones, requiring no additional hardware.

## Special Input Controls

AfterSchool Ally does not require custom input controls, relying on standard UI controls for simplicity and accessibility.

### **Discussion**:

* **Standard Controls**: The platform uses common controls like text fields (for registration), dropdown menus (for activity types), buttons (for submitting forms), and calendar pickers (for selecting dates). These are easily implemented in React and Tailwind CSS, aligning with user familiarity.
* **Multilingual Support**: Includes a language switcher (dropdown for English, Spanish, Chinese) to ensure accessibility for diverse low-income communities.
* **Rationale**: Avoiding complex controls (e.g., gesture-based or voice inputs) reduces development costs and learning curves, suiting non-technical users (parents, volunteers).
* **Optional Controls**: Future consideration for simple drag-and-drop interfaces (e.g., for volunteer scheduling), but standard controls are prioritized for compatibility and ease of use.

## Contextual Constraints and Advantages

### **Constraints**:

* **Device Limitations in Low-Income Families**: Users may have low-performance or older devices, requiring optimized load times and resource usage (e.g., compressed images, minimal JavaScript).
* **Internet Connectivity Issues**: Low-income communities may face unstable networks, necessitating offline capabilities (e.g., caching event details) and low-bandwidth optimization.
* **Language Diversity**: Diverse communities require multilingual support, increasing translation and localization costs.
* **Data Privacy**: Storing children’s and donors’ data must comply with regulations (e.g., GDPR, COPPA), adding development complexity.

### **Advantages**:

* **Cloud Deployment**: Using cloud services (e.g., AWS, Firebase) ensures scalability and high availability, reducing maintenance costs.
* **Multi-Platform Access**: Web and mobile compatibility support a broad user base, enhancing accessibility.
* **AI Integration Potential**: AI can optimize volunteer matching and report generation (e.g., recommendation algorithms, automated charts), improving efficiency (see AfterSchool HQ’s similar approach).
* **Nonprofit Focus**: Free services attract low-income families and funders, boosting community support and donation potential.

# Personas

Develop three persona descriptions to represent the target market for the application. They should apply to the application, but also be rather distinct from each other.

## A person smiling at the camera AI-generated content may be incorrect.Persona 1: Parent - Maria Gonzalez

* **Background**:
  + Age: 35
  + Occupation: Supermarket cashier, low-income single mother
  + Location: Low-income urban community, USA
  + Tech Level: Basic smartphone use (Android 8.0), familiar with WhatsApp and simple apps
  + Language: Primarily Spanish, limited English
* **Needs and Goals**:
  + Find free after-school programs (e.g., math tutoring, art activities) for her 9-year-old daughter to support academics.
  + Needs flexible activity schedules to fit shift work.
  + Wants real-time notifications (e.g., event reminders, daughter’s attendance).
  + Requires a Spanish interface for easy navigation.
* **Relation to AfterSchool Ally**:
  + Uses the event management module to browse and register for activities, filtering by age or interest.
  + Accesses the parent communication module to check the daughter’s homework and teacher feedback.
  + Relies on multilingual support (Spanish) to use the platform comfortably.
* **Behaviors**:
  + Checks the mobile app weekly to register for 2-3 activities.
  + Enables notifications to stay updated on daughter’s activities.

## A person smiling at camera AI-generated content may be incorrect.Persona 2: Volunteer - Li Wei

* **Background**:
  + Age: 22
  + Occupation: Third-year education major
  + Location: Suburban area near a community center
  + Tech Level: Proficient with smartphones and web apps, active on social media
  + Language: Fluent in Chinese and English
* **Needs and Goals**:
  + Seeks volunteer opportunities to gain teaching experience and enhance resume.
  + Needs flexible volunteering hours to fit academic schedule.
  + Wants recognition (e.g., certificates or badges) to stay motivated.
* **Relation to AfterSchool Ally**:
  + Registers skills (e.g., math tutoring, art instruction) and availability in the volunteer coordination module.
  + Receives matched activity assignments and views event details.
  + Earns “contribution badges” to boost engagement.
* **Behaviors**:
  + Volunteers for 2-4 activities monthly, prioritizing education-related tasks.
  + View the contribution history on the platform, and share badges on social media.

## A teacher reading a book to students AI-generated content may be incorrect.Persona 3: Admin - Sarah Thompson

* **Background**:
  + Age: 40
  + Occupation: Nonprofit project coordinator
  + Location: Urban area, managing local after-school programs
  + Tech Level: Familiar with office software and basic management tools (e.g., Google Workspace)
  + Language: Native English
* **Needs and Goals**:
  + Efficiently manage afterschool activities (e.g., scheduling, volunteer allocation, attendance tracking).
  + Needs impact reports to attract foundation and corporate funding.
  + Wants to streamline administrative tasks to focus on community service.
* **Relation to AfterSchool Ally**:
  + Uses the event management module to create and publish activities, track registrations, and monitor attendance.
  + Tracks donations and generates visualized reports (e.g., attendance charts) via the donor tracking module.
  + Assigns tasks to volunteers through the coordination module to ensure smooth operations.
* **Behaviors**:
  + Logs in weekly to manage 10-15 activities and assign volunteers.
  + Generates monthly reports to showcase outcomes to funders.

# Use Cases

Develop a fairly complete listing of the use cases for functional requirements for this app and create **a diagram for them**. This list should **provide a name for each use case and a brief description** (one or two sentences). Then, **from that list, choose two** of the more interesting use cases, and develop use case descriptions for them. Don’t choose things like “login” use cases, as they are not specific to this app and provide little insight into how the app works.

The two (or more) use cases you chose to pursue will be the ones shown working in the prototype.

## Use Case List

Below is a fairly complete list of use cases for AfterSchool Ally’s functional requirements, each with a name and brief description (one or two sentences):

1. **Register for Activity**  
   Parents browse and filter activities by child’s age or interest through the event management module, then submit a registration request and receive a confirmation notification.
2. **Coordinate Volunteers**  
   Admins use the system to match registered volunteers to specific activities based on their skills and availability.
3. **Earn Points**  
   Children earn points by completing activities or homework, which can be viewed and redeemed for rewards (e.g., stationery or priority registration) on the platform.
4. **View Feedback**  
   Parents log in to check their child’s attendance records, homework completion, and personalized feedback from teachers.
5. **Generate Impact Report**  
   Admins input donation and activity data, and the system automatically creates visualized reports (e.g., attendance charts) to attract funding.
6. **Adjust Activity Schedule**  
   Admins modify activity times based on registration or volunteer feedback, notifying relevant parents and volunteers.
7. **Claim Contribution Badge**  
   Volunteers receive “contribution badges” from the system after completing activities, which they can view or share on the platform.
8. **Receive Real-Time Notifications**  
   Parents and volunteers get instant updates via mobile devices, such as activity changes, registration confirmations, or task assignments.
9. **Track Donations**  
   Admins record donation sources and amounts, generating personalized thank-you letters for donors.
10. **Recommend Activities**  
    The system suggests future activities for parents based on their child’s past participation and interests.

## Detailed Use Case Descriptions

The following two interesting use cases are selected for detailed descriptions and will be demonstrated in the prototype:

### Use Case 1: Earn Points

* **Use Case Name**: Earn Points
* **Actors**: Child (via Parent operation), System
* **Preconditions**: Child is registered and has participated in an activity; parent is logged into the platform.
* **Basic Flow**:
  1. Child completes an activity or homework (e.g., math tutoring or art workshop).
  2. Volunteer or admin confirms participation and submits a completion record on the platform.
  3. The system automatically adds points to the child’s account (e.g., 5 points per hour).
  4. Parent logs in to view the child’s point balance and leaderboard ranking.
  5. Parent selects to redeem points for rewards (e.g., stationery or priority registration).
* **Alternative Flow**:
  1. If a volunteer fails to confirm, the parent can submit proof of participation (e.g., a photo), and an admin manually reviews and adds points.
* **Postconditions**: Child’s account updates with the new point balance; parent receives a redemption success notification.
* **Exceptions**: If the activity is incomplete, no points are awarded, and the parent is notified of the reason.
* **Frequency**: Approximately 10-20 times per week, depending on activity volume.

### Use Case 2: View Feedback

* **Use Case Name**: View Feedback
* **Actors**: Parent, System
* **Preconditions**: Child has participated in an activity, the teacher has submitted feedback, and the parent is logged into the platform.
* **Basic Flow**:
  1. Parent logs in and selects the “View Feedback” option.
  2. System displays the child’s recent activity attendance, homework completion, and teacher comments (e.g., “Worked hard on math homework”).
  3. Parents can filter feedback by specific activities or date ranges for detailed views.
  4. Parent saves or downloads the feedback record for family discussions.
* **Alternative Flow**:
  1. If feedback is not updated, the system prompts the parent to contact the admin, who then requests the teacher to provide input.
* **Postconditions**: Parent accesses the latest feedback; system logs the viewing history.
* **Exceptions**: If the system fails, the parent can contact support to obtain a feedback copy.
* **Frequency**: Approximately 5-10 times per week, depending on activity participation.

# Activity Diagram

Develop an activity diagram for **one** of the use cases that you are prototyping.

The following activity diagram is developed for the use case “**Earn Points**,” illustrating the process of children earning points by completing activities. This diagram is tailored for prototype development, highlighting interactions among parents, volunteers, admins, and the system.

* **Initial Node**: The process starts with “Child Begins Activity.”
* **Main Activity Flow**:
  1. Child completes an activity or homework (e.g., math tutoring or art workshop).
  2. Volunteer or admin confirms the child’s participation and submits a completion record.
  3. System validates the record (e.g., checks activity time and proof of participation).
  4. If valid, the system automatically adds points (e.g., 5 points per hour) to the child’s account.
  5. Parent logs in to view the point balance and leaderboard ranking.
  6. Parent selects to redeem points for rewards (e.g., stationery or priority registration).
  7. System updates the point balance and sends a redemption success notification.
* **Decision Node**:
  1. After step 3, if the record is invalid (e.g., missing confirmation), the flow branches to “Parent Submits Evidence.”
  2. Admin reviews the evidence; if approved, returns to step 4 to add points; if rejected, the process ends with no points awarded.
* **Final Node**: The process ends after “Notify Parent of Redemption Result.”
* **Concurrent Region** (Optional): Volunteers and admins can confirm records simultaneously (indicated by Fork/Join), improving efficiency.

## Diagram Screenshot:

[Link Here](https://lucid.app/lucidchart/69a179e9-dee2-45de-b1a9-d2f8f8db9a86/edit?invitationId=inv_726a1984-3890-4fda-b718-2b5654bc191a)

Also, in **/diagrams/Activity Diagram.png**

A diagram of a child's activity

AI-generated content may be incorrect.

# Entities

Provide a fairly complete list of business entities for the application. Identify some attributes for each entity. A tabular format would work well for this.

Create an ERD for this application that shows entities, attributes, relationships and cardinality.

|  |  |
| --- | --- |
| Entity Name | Attributes |
| User | UserID, Name, Email, Password, Role (Parent/Volunteer/Admin), Language Preference, Registration Date |
| Child | ChildID, Name, Age, ParentID, Grade, Interests (multi-select), Points Balance |
| Activity | ActivityID, Name, Description, DateTime, Location, Capacity, Status (Open/Full/Canceled), Category |
| Registration | RegistrationID, UserID, ActivityID, Registration Date, Status (Pending/Confirmed/Canceled), Attendance Record |
| Feedback | FeedbackID, ChildID, ActivityID, TeacherID, Comment, Submission Date, Rating (1-5) |
| Volunteer | VolunteerID, UserID, Skills (multi-select, e.g., Math Tutoring/Art), Availability, Contribution Badge Count |
| Donation | DonationID, Donor Name, Amount, Date, Status (Pending/Completed), Notes |
| Points | PointsID, ChildID, ActivityID, Earned Date, Points Amount, Redemption Status (Unredeemed/Redeemed) |
| Report | ReportID, Generation Date, Data Range (Start/End Date), Type (Attendance/Donation), File Path, DonationID (optional) |

## ERD Diagram:

(Recommending looking through the link below due to the messy relations in the screenshot)

[Link here](https://lucid.app/lucidchart/dac63f42-b5fa-4766-9475-d293acd1ce2e/edit?viewport_loc=-1810%2C-200%2C3451%2C2758%2C0_0&invitationId=inv_2148c346-840e-4813-9461-b02c09243726)

Also, it can be found in: **/diagrams/ERD.png**

A screenshot of a computer

AI-generated content may be incorrect.

# Content Plans

Provide a description of the text, features and input controls on each page being prototyped.

Below are the content plans for the prototyped pages of the AfterSchool Ally application, covering the “Earn Points” and “View Feedback” use cases. These descriptions outline the text, features, and input controls for each page to enhance the user experience for parents and children.

## Page 1: Earn Points

* **Text**:
  + Title: “My Points”
  + Description: “Check your child’s point balance and leaderboard, redeem rewards!”
  + Status Message: “Your child’s points have been updated” or “Please confirm activity completion to earn points.”
  + Points Details: “Current Points: 50 | Leaderboard Rank: 3rd”.
* **Features**:
  + Display the child’s point balance and leaderboard ranking.
  + Offer options to redeem points (e.g., stationery or priority registration).
  + Allow parents to submit evidence (e.g., photos) for manual point requests.
* **Input Controls**:
  + Dropdown Menu: Select a reward to redeem (options: Stationery, Priority Registration).
  + Upload Button: Submit evidence of participation (file types: JPEG/PNG).
  + Confirm Button: Submit the redemption request.
  + Refresh Button: Update points and leaderboard data.

## Page 2: View Feedback

* **Text**:
  + Title: “Feedback Records”
  + Description: “View your child’s recent activity feedback and attendance records.”
  + Feedback Content: “Math Tutoring: Worked hard on homework, Rating: 4/5” or “No recent feedback.”
  + Date Label: “Submission Date: 2025-07-28”.
* **Features**:
  + Display the child’s attendance records, homework completion, and teacher comments.
  + Enable filtering of feedback by activity or date range.
  + Provide an option to download feedback records as a PDF.
* **Input Controls**:
  + Dropdown Menu: Select an activity name (loaded from database).
  + Date Picker: Filter feedback by date range (start/end dates).
  + Download Button: Export feedback as a PDF file.
  + Contact Button: Send a feedback request to the admin.

# Prototypes

Develop professional mockups of the screens for the two (or more) use cases you have chosen to develop using JustInMind Prototyper or **Figma**.

## Screenshot below

Also, can find in: **/diagrams/Prototype.fig**

Screens screenshot of a screen

AI-generated content may be incorrect.

# Class Diagram

Using a UML modelling tool such as Visual Paradigm, LucidChart or Mermaid, develop a class diagram for the application. It should include:

At least 5 classes with appropriate names (but you may find you need quite a few more)

* Attributes
* Methods
* Visibility (privacy) of classes, attributes and methods
* Associations between classes
* Inheritance (think, for example, of Users of all kinds)

[Link here](https://lucid.app/lucidchart/3be60471-c467-4e3f-ba5b-f26ae5749495/edit?viewport_loc=-1551%2C-819%2C1977%2C2088%2C0_0&invitationId=inv_64c97598-80eb-4484-8e34-12baa485e6d8)

## Screenshot below

A computer screen shot of a computer

AI-generated content may be incorrect.

# Implementation - Developing the Agile Plan

You are developing a plan to implement the system in an **agile, iterative, scrum** manner. **Iterative** means developing the system in planned stages in which use cases are developed several at a time, as long as they can be done concurrently.

**Agile** means that changes can be made quickly and easily to the plan depending on new information about system requirements that has arisen, or else as a result of changes in schedule (due to environmental factors, availability of human resources and the client, etc.). **Scrum** means you are using the specific methodology in which development is done in **sprints,** and developers, a **product owner,** and a **scrum master** follow the best practices of scrum development. \*\***Note** that **Scrum** implies **iterative** and **agile** development, but the reverse may not always be true. We are using **Scrum** development. Make an assumption that this development for this site will have three **sprints** (implementation stages) of three weeks each. In a spreadsheet template provided for this purpose, record the Agile plan for implementation that you have developed.