CAMPUSEXPENSE MANAGER MOBILE APPLICATION

GROUP 1
PRESENTATION

Project Overview

BudgetWise Solutions is developing the CampusExpense Manager mobile application to assist university students in effectively managing their expenses and staying within their budgets. The app is designed to simplify expense tracking for students living both on and off campus, empowering them to make informed financial decisions

Objectives of the App

The CampusExpense Manager app empowers university students to manage their finances effectively. It allows users to set and track budgets across various expense categories, promoting responsible spending and clearer financial awareness. With an intuitive interface for logging expenses and customizable categories, students can tailor their financial tracking. Data visualization features, like charts, help identify spending trends, while notifications alert users when approaching budget limits. Security is ensured through encryption and hashing techniques to protect sensitive information. The app also functions offline, making it accessible in areas with limited connectivity. By integrating user feedback and offering educational resources, the CampusExpense Manager supports both expense tracking and financial literacy

User Requirements

- **Expense Tracking**

Users need a straightforward method to log and categorize their expenses effortlessly. This includes adding details such as amount, description, date, and category for each expense.

- **Budget Alerts**

The app must provide alerts and notifications when users approach or exceed their set budget limits for various categories. This feature is crucial for helping users manage their finances proactively.

- **Intuitive User Interface**

A simple and user-friendly interface is essential for ensuring that users can navigate the app easily, allowing for quick entry and retrieval of financial information without confusion.

- **Categorization of Expenses**

Users require the ability to categorize expenses into predefined and custom categories (e.g., groceries, rent, entertainment) to better analyze their spending patterns.

- **Summary Reports**

The app should offer summary reports that provide an overview of spending habits, including total expenses, remaining budget, and insights into expenditure trends over time. This feature helps users reflect on their financial behavior and make informed decisions.

Systems Investigation and Research

Competitor Analysis

An analysis of existing expense management apps revealed several strengths and weaknesses:

- Strengths: Many competitors offered robust features like budget tracking, expense categorization, and reporting. User-friendly interfaces were common among the top apps.
- Weaknesses: Some apps lacked customization options and failed to provide adequate notifications for budget limits. Others had complex navigation, making them less appealing to students.

Insights into Student Needs

Through our research, we identified key features that students required in an expense management app:

- Simplicity: Users emphasized the need for an intuitive interface that allows for quick and easy expense entry.
- Budget Alerts: Students expressed a strong desire for real-time notifications when nearing budget limits to help them stay financially responsible.
- Customization: The ability to create custom categories for expenses was highlighted as essential for accurately tracking diverse spending habits.
- Data Visualization: Many students preferred visual representations of their finances, such as charts and graphs, to better understand their spending patterns.

Project Scope and Constraints

Project Scope

The CampusExpense Manager app aims to provide a comprehensive financial management tool for university students. The scope includes:

- Core Features:
  - User Registration and Authentication: Secure account creation and login.
  - Expense Tracking: Log and categorize expenses with details like amount, date, and description.
  - Budget Setting and Alerts: Set monthly budgets and receive notifications for limits.
  - Expense Categorization: Predefined and customizable categories for tracking.
  - Summary Reports and Data Visualization: Insights into spending patterns through charts.
  - Recurring Expenses Management: Easily manage recurring expenses.
  - Offline Functionality: Operate without internet access for accessibility.
  - Cloud Synchronization: Sync data to the cloud when online for consistency.
- Target Audience:
  - Primarily aimed at university students seeking effective personal finance management.

Project Scope and Constraints

Limitations

- Budget Constraints:
  - Limited budget restricting marketing and advanced features, necessitating cost-effective solutions.
- Team’s Technical Experience:
  - Junior developers with moderate experience may limit feature complexity and technology use.

Constraints Influencing Design Choices

- Data Privacy Compliance:
  - Adhering to regulations affects authentication methods and data storage practices.
- Offline Functionality:
  - Designing for offline use influences storage and synchronization strategies.
- User Feedback Integration:
  - Incorporating feedback for improvements may impact the development timeline.
- Cross-Platform Compatibility:
  - Development for both Android and iOS increases complexity and resource needs.
- Performance Optimization:
  - Ensuring a smooth user experience influences design choices in data handling.
- Limited Marketing Resources:
  - Constrained budget for marketing limits outreach strategies.

Tools and Technologies Used

The Tech Stack Behind the App Key Tools and Technologies:

Firebase:

Realtime Database: For seamless data synchronization across devices.

Authentication: To secure user login and data access.

Cloud Functions: For serverless backend logic and automated tasks.

Cloud Storage: To store user-generated content and app resources.

Android Studio:

The official IDE for Android app development.

Provides a comprehensive set of tools for building, testing, and debugging.

Java/Kotlin:

Programming languages used to build Android apps.

Kotlin's concise syntax and modern features enhance development efficiency.

Figma:

Design tool for creating user interfaces and prototypes.

Enables collaboration and iterative design.

Why We Chose These Tools:

Real-time features for instant updates.

Cost-effective solution for startups.

Comprehensive development environment.

Mature languages with extensive libraries and frameworks.

Easy-to-use interface and powerful features.

Development Methodology

We chose Agile because :

Adaptability: Agile's iterative approach allows us to respond quickly to changing requirements and emerging technologies.

Collaboration: Close collaboration between the development team and stakeholders ensures alignment and timely feedback.

Customer Focus: By delivering working software frequently, we can gather valuable user feedback and prioritize features accordingly.

Risk Mitigation: Agile's iterative nature helps identify and address potential risks early on.

Our Agile Process:

Planning:

- Define the project scope and goals.
- Create a high-level project plan.

Design:

- Design the user interface and user experience.
- Create wireframes and prototypes.

Development:

- Implement features in short development cycles (sprints).
- Conduct regular code reviews and testing.

Testing:

- Test the app for functionality, performance, and security.
- Gather user feedback and iterate on the design.

Deployment:

- Deploy the app to app stores.
- Monitor app performance and user feedback.

Initial Design and Prototyping

Our Design Process:

User Research:

- Conducted interviews and surveys to understand user needs and pain points.
- Identified key features and functionalities.

Wireframing:

- Created low-fidelity wireframes to visualize the app's layout and information hierarchy.
- Focused on the core user flows and interactions.

Prototyping:

- Developed high-fidelity prototypes using Figma to simulate the app's look and feel.
- Incorporated visual design elements and interactive features.

User Testing:

- Conducted usability tests with potential users to gather feedback on the prototype.
- Identified areas for improvement in terms of usability and user experience.

Key Design Considerations:

- Intuitive Interface: A clean and intuitive interface that is easy to navigate.
- Visual Clarity: Clear and concise visual elements to enhance user understanding.
- Mobile-First Approach: Optimized for mobile devices to ensure a seamless user experience.
- Offline Functionality: Designed to work offline, allowing users to track expenses even without an internet connection.

User Interface (UI) Design

Key Design Principles:

Simplicity: A minimalist approach to reduce cognitive load and improve usability.
Intuitiveness: Self-explanatory design elements that guide users effortlessly.
Consistency: Consistent use of typography, color, and layout for a cohesive user experience.
Accessibility: Adherence to accessibility guidelines to accommodate users with disabilities.

Design Rationale:

Clean and Minimalist Interface: A clutter-free design focuses attention on essential information.
Clear and Concise Labels: Easy-to-understand labels and descriptions for all elements.
Visual Hierarchy: Effective use of typography and spacing to prioritize information.
Intuitive Navigation: Simple navigation patterns that guide users through the app.
Consistent Visual Language: A consistent visual style throughout the app for a unified experience.

Backend Architecture

Firebase as the Core Backend Solution:

- **Firebase Firestore:**
  - **NoSQL Database:** Stores user data in a flexible, scalable format. This is particularly useful for managing diverse expense data and user profiles.
  - **Real-time Data Sync:** Automatically updates data across all connected clients, ensuring users see the most current information without needing to refresh.
- **Firebase Authentication:**
  - **Secure User Authentication:** Supports various authentication methods (email/password, Google, Facebook), allowing users to choose their preferred login method.
  - **Easy Management:** Provides built-in user management features, simplifying account creation, verification, and password recovery.

Support for Scalability:

- **Automatic Scaling:** Firebase's infrastructure scales automatically to accommodate growing data needs and user traffic without manual intervention.
- **High Availability:** Firebase is designed to be reliable and maintain uptime, crucial for an app that students rely on for financial management.
- **Cost-Effective:** Pay-as-you-go pricing model allows BudgetWise Solutions to manage costs effectively as the user base grows.
- **Extensibility with Cloud Functions:** Firebase Cloud Functions enable the addition of server-side logic, such as processing complex queries or sending notifications, without the need for a dedicated server, further supporting scalability and flexibility.

Core Features Implemented

1. **Expense Logging**

- **Feature Overview:**
  - Users can log expenses through a straightforward interface, entering details such as:
    - **Description**: A brief title or note about the expense (e.g., "Grocery shopping").
    - **Amount**: The total cost of the expense.
    - **Date**: The date the expense was incurred.
    - **Category**: A dropdown menu for selecting a predefined category or creating a new one.
- **User Requirement Fulfillment:**
  - **Ease of Use:** The logging process is streamlined to minimize the time taken to record expenses, catering to students' fast-paced lifestyles.
  - **Comprehensive Tracking:** By allowing users to log all types of expenses, the app ensures a complete view of their financial activity, aiding in better financial management.
  - **Recurring Expense Options:** Users can mark certain expenses as recurring (e.g., rent), which will automatically populate in future budget periods, reducing manual entry.

2. Category Selection

- **Feature Overview:**
  - Users can categorize their expenses into different groups such as:
    - Predefined Categories: Common categories like "Food," "Transportation," "Entertainment," "Housing," etc.
    - Custom Categories: Users can create their own categories based on personal spending habits (e.g., "Books," "Clothing").
- **User Requirement Fulfillment:**
  - **Organized Tracking:** Categorization allows users to visualize spending patterns, making it easier to see where their money is going.
  - **Personalization:** The ability to create custom categories ensures that the app meets individual needs, enhancing user engagement and satisfaction.
  - **Enhanced Budgeting:** Users can set specific budgets for each category, helping them allocate funds more effectively.

Core Features Implemented

3. Summary View

- **Feature Overview:**
  - A dashboard that aggregates and displays key financial information, including:
    - **Total Spending:** A cumulative total of all expenses logged for the current month.
    - **Remaining Budget:** An overview of how much budget is left in each category.
    - **Visual Breakdown:** Charts (e.g., pie charts, bar graphs) that illustrate spending by category and compare budgeted versus actual spending.
- **User Requirement Fulfillment:**
  - **Financial Insights:** The summary view provides a clear snapshot of financial health, helping users quickly assess their spending habits.
  - **Budget Management:** By comparing actual spending against set budgets, users can identify overspending in specific areas and make informed adjustments.
  - **Trends Over Time:** Users can view historical data to track their expenses over different months, enabling them to recognize trends and adjust their spending behavior accordingly.

Additional Features

1. **Manual Categorization**

- **Feature Overview:**
  - **Custom Category Creation:** Users can create new categories on-the-fly, allowing them to label expenses in a way that resonates with their specific financial habits. For instance, if a user frequently spends on "Coffee Shops," they can create that category for better tracking.
  - **Editing Existing Categories:** Users can modify or delete categories to keep their expense tracking relevant over time.
  - **User Interface:** The feature includes a user-friendly modal or pop-up that allows easy entry and modification of categories.
- **User Requirement Fulfillment:**
  - **Personalization:** This feature caters to diverse spending behaviors, accommodating students who may have unique financial needs based on their lifestyle, major, or personal interests.
  - **Detailed Insights**: By enabling users to categorize expenses granularly, the app can provide more detailed analytics, such as monthly spending patterns in specific areas, helping users identify trends and adjust their budgets accordingly.

2. Notifications for High Spending

- **Feature Overview:**
  - **Budget Threshold Alerts:** Users can set customizable thresholds for spending notifications. For example, they might choose to receive alerts when they reach 75% of their budget for "Food."
  - **Instant Notifications:** The app uses push notifications to alert users in real-time when they are nearing their budget limits.
  - **Daily/Weekly Summaries:** In addition to real-time alerts, users can opt in for daily or weekly summaries of their spending to keep them informed about their financial status.
- **User Requirement Fulfillment:**
  - **Proactive Management:** These notifications help users take immediate action if they are overspending, allowing them to adjust their habits before it's too late.
  - **Behavioral Change:** Regular alerts can encourage users to be more mindful of their spending, ultimately leading to better financial decisions and greater accountability in managing their budgets.

Additional Features

Prioritization Based on User Feedback

- **User-Centric Development:**
  - **Feedback Collection:** During the alpha and beta testing phases, BudgetWise Solutions conducted surveys and focus groups to gather insights from potential users about desired features.
  - **Top Requests:** Manual categorization and high-spending notifications emerged as two of the most requested features, with users expressing a strong desire for increased control over their financial tracking and proactive alerts to manage their budgets better.
- **Iterative Approach:**
  - **Agile Development:** The team employed agile methodologies, allowing for rapid iterations based on user feedback. This included:
  - **Prototyping:** Creating mockups of the categorization interface and notification settings, which were tested with users for feedback before final implementation.
  - **User Testing:** Conducting usability tests to ensure that the features were intuitive and met user expectations.
- **Future Enhancements:**
  - **Enhanced Reporting Features:** Users have expressed interest in generating customizable reports that break down expenses by category or time period, allowing them to analyze their spending more effectively.
  - **Community Features:** Potential future features could include forums or social sharing options, where users can share budgeting tips or successes with peers, fostering a community of financially responsible students.

Data Management and Security

1. **Data Management**

- **Firebase Firestore:**
  - **Structured Data Storage:**
    - **NoSQL Database:** Firestore uses a document-oriented database model, allowing for nested data structures that fit the app's needs. Each user's data is stored in separate documents within collections, making it easy to manage and scale.
    - **User Document Structure:** Each user document might contain sub-collections for expenses, budgets, and categories, enabling efficient grouping and retrieval of related data.
  - **Real-Time Synchronization:**
    - **Immediate Updates:** Any change made to the user's data (e.g., logging a new expense) is instantly synchronized across all devices. This feature is crucial for users who may access the app from multiple devices (e.g., phone, tablet, laptop).
    - **Offline Capabilities:** Firestore supports offline data persistence, allowing users to log expenses without an internet connection. Data is queued and synchronized once connectivity is restored.
  - **Data Querying:**
    - **Powerful Querying:** Users can perform complex queries to retrieve data based on multiple parameters (e.g., filter expenses by date range and category). This helps users analyze their spending patterns effectively.
    - **Indexing:** Firestore automatically indexes data to optimize query performance, ensuring quick data retrieval even as the volume of data grows.

Data Management and Security

2. Data Security

- **Firebase Authentication:**
  - **User Authentication:**
    - **Multiple Sign-In Options:** Users can register and log in using various methods, including email/password, Google, and Facebook. This reduces the friction in the onboarding process and enhances user convenience while maintaining security.
    - **Password Recovery:** Firebase provides built-in support for password reset, allowing users to securely regain access to their accounts if they forget their passwords.
  - **Identity Verification:** Email verification is required for new accounts, adding an extra layer of security to confirm user identities and prevent fake accounts.
- **Data Protection Features:**
  - Role-Based Access Control:
    - **User Data Isolation:** Firestore security rules ensure that each user can only access their own data. For example, if a user tries to access expense data stored under another user's account, the request is denied.
    - **Custom Security Rules:** Developers can define granular rules that specify who can read or write to specific documents or collections. For example, only authenticated users can create new expense entries, while everyone can read general tips or educational content.
  - **Data Encryption:**
    - **Encryption at Rest:** All data stored in Firestore is encrypted using industry-standard methods, protecting it from unauthorized access even if physical security is compromised.
    - **Encryption in Transit:** Data transmitted between the app and Firestore is encrypted using SSL/TLS protocols, ensuring that user data cannot be intercepted during transmission.

Data Management and Security

3. Compliance and Best Practices

- **GDPR Compliance:**
  - **User Rights:** The app provides mechanisms for users to request their data, delete their accounts, or export their information, complying with GDPR regulations.
  - **Transparent Data Practices:** The app includes a privacy policy that outlines how user data is collected, used, and protected, fostering trust with users.
- **Regular Security Audits:**
  - **Vulnerability Assessments:** Firebase conducts regular security assessments to identify and mitigate potential vulnerabilities, ensuring that the platform remains secure against evolving threats.
  - **Updates and Patches:** Firebase continuously releases updates and patches to address security issues, which are applied automatically to ensure the latest security measures are in place.

4. Backup and Recovery:

- **Automated Backups:**
  - **Daily Backups:** Firebase provides automated daily backups of user data, ensuring that even in the event of a data loss incident, recent data can be restored.
  - **Versioning:** Firestore supports versioning, allowing users to retrieve previous versions of their data if needed.
- **Disaster Recovery:**
  - **Redundant Infrastructure:** Firebase operates on a globally distributed infrastructure, ensuring high availability and reliability. In the event of a server failure, user data remains accessible through redundant systems.
  - **Disaster Recovery Protocols:** Firebase has established protocols for disaster recovery, minimizing downtime and ensuring rapid restoration of services.

User Testing and Feedback

The user testing process typically includes the following key steps:

- Planning: Define the testing goals, target audience, and testing scenarios.
- Designing scenarios: Create scenarios or tasks for users to complete, which helps evaluate the product's usability and effectiveness.
- Selecting participants: Choose users who represent the target audience. This usually involves 5 to 10 people.
- Conducting the test: Observe users as they perform tasks. Record their feedback and behaviors.
- Analyzing data: Review the collected results to identify issues and strengths of the product.
- Reporting findings: Create a report detailing the findings, recommendations, and suggestions for product improvement.
- Improving the product: Based on user feedback, make necessary adjustments to enhance the user experience.

User Testing and Feedback

User Testing Insights:

Intuitive Navigation: Users appreciated the app's simple and intuitive navigation.

Clear Information Display: The clear and concise presentation of expense data was well-received.

Quick and Easy Expense Entry: Users found the expense entry process efficient and straightforward.

Feedback and Improvements:

Enhanced Search Functionality: Implemented a more robust search feature to quickly find specific expenses.

Improved Budget Visualization: Enhanced the visual representation of budgets and spending limits.

Simplified Settings Menu: Streamlined the settings menu for easier access to important options.

Challenges Faced

During the process of completing the project, the team encountered many challenges.

- There were technical difficulties, the members still lacked a lot of knowledge and experience about the programming language being used to do the project.
- There were some problems with the equipment used to run the software for the project (such as: the device when running the software was jerky, laggy, frozen. The software had errors and could not run).
- Regarding teamwork, members are still not able to fully connect with each other, it is still difficult to find a common voice in the process of developing project software.
- The group is having difficulty working as a team, which leads to the group losing direction, causing the project to be completed on time and possibly delayed compared to the expected deadline.

Challenges Faced

- To overcome the lack of understanding of the language to do the project, the group sought to learn from people who had experience and knowledge of that language and referred to similar products to the project the group was working on so that they could rely on and apply them to the group's project.
- Regarding equipment issues, everyone in the group will support the member whose equipment is having problems as quickly as possible so that the project can continue to be implemented.
- Regarding teamwork, the members sat down to discuss with each other and found a common voice and were also able to agree on the next direction in the group project to be able to meet the completion schedule.

Quality Assurance and Testing

Our Testing Process:

Functional Testing:

- Verifying that all features and functionalities work as expected.
- Testing various scenarios, including edge cases and error conditions.

Usability Testing:

- Observing real users interacting with the app to identify usability issues.
- Gathering feedback on the app's ease of use and overall experience.

Performance Testing:

- Assessing the app's performance under different load conditions.
- Optimizing the app for speed and responsiveness.

Security Testing:

- Identifying and addressing potential security vulnerabilities.
- Ensuring data privacy and protection.

Key Testing Results:

High-Quality User Experience:

- Minimal bugs and crashes.
- Intuitive and easy-to-use interface.

Robust Performance:

- Smooth and responsive performance, even under heavy load.
- Efficient resource utilization.

Enhanced Security:

- Strong security measures to protect user data.
- Adherence to industry best practices.

Risk Management

Identified Risks:

Limited Mobile Development Experience: The team's limited experience in mobile app development could impact the project timeline and quality.
Scope Creep: Unforeseen requirements or feature additions could lead to project delays and budget overruns.

Mitigation Strategies:

Skill Development:
Provided extensive training and mentorship to team members.
Encouraged continuous learning and skill improvement.

Agile Methodology:

- Adopted an Agile approach to adapt to changing requirements and prioritize features.
- Regular reviews and iterations to ensure project focus.

Clear Communication:

- Established effective communication channels between the development team and stakeholders.
- Clearly defined project scope and expectations.

Regular Progress Reviews:

- Conducted regular project reviews to track progress and identify potential risks.
- Implemented a rigorous change management process.

Tools and Strategies for Effective Team Collaboration

Project Management Tools

Trello: Used for task assignment, tracking progress, and ensuring deadlines are met.

GitHub: Version control and collaborative coding to streamline development workflows.

Slack: Facilitates real-time communication within the team.

Google Drive: Centralized repository for design mockups, documentation, and resources.

Tools and Strategies for Effective Team Collaboration

Collaboration Strategies

Agile Methodology: Weekly sprints with clear goals, deliverables, and retrospectives.
Daily Standups: 15-minute check-ins to address roadblocks and align on priorities.
Peer Code Reviews: Enhance code quality and provide mentorship for junior developers.
Mentorship Sessions: Senior developers provide training on mobile development best practices.
Feedback Loop: Regular check-ins with stakeholders to ensure alignment with user needs.

Metrics to Evaluate Project Success

Functionality:
All core features (expense tracking, budget setting, added transaction) are implemented and functional.

User Requirements Alignment:
App meets functional and non-functional requirements, including offline capability and platform compatibility.

User Satisfaction:
User surveys and app ratings post-launch.

Performance:
App load time and responsiveness tested under high data volumes.

Security Compliance:
Adherence to data privacy regulations (GDPR).

Error Rates:
Frequency of bugs or crashes, aiming for less than 2% crash rate.

Lesson Learned

Delivering on Promises: User-Centric Features

User Registration and Authentication: Implemented secure login and account creation.
Expense Tracking: Users can add, edit, and categorize expenses.
Budget Setting: Customizable monthly budgets for expense categories.
Expense Overview: Summary screens showing total spending and remaining budgets.
Recurring Expenses: Functionality to manage and automate recurring expenses.
Expense Notifications: Alerts for budget limits approached or exceeded.

Lesson Learned

Key Takeaways from the Development Journey

Technical Skills:
Mastery of cross-platform development frameworks
Understanding secure data handling and encryption methods.

Team Collaboration:
Improved coordination through agile workflows and communication tools.

User-Centered Design:
Importance of iterative user testing to refine the interface.

Challenges Overcome:
Navigating platform compatibility and offline-first design.
Budget constraints fostering innovative problem-solving.

Continuous Growth through Feedback

Post-Launch Feedback:
Users praised intuitive UI and offline capability.
Suggestions included integration with bank accounts and advanced analytics.

Actions Taken:
Fixed minor bugs based on user reports.
Enhanced notification settings for better user experience.
Implemented a feedback form for easier user input.

Future Improvements:
Explore in-app financial tips for students.
Add multi-language support.
Evaluate monetization strategies that align with user preferences.
Exploration of gamification elements to encourage budget adherence.

Feedback Analysis and Future Improvements

Feedback Overview

Positive Feedback

- Users appreciated the intuitive interface and ease of expense tracking.
- The budget alert feature was praised for helping users manage their finances effectively.
- Data visualization tools, such as charts, were well-received for providing clear insights into spending habits.

Constructive Feedback

- Some users reported difficulties with the categorization of expenses, indicating a need for more customizable options.
- Requests for additional filtering options in expense reports were common, particularly to view data over specific time frames.
- Users expressed a desire for enhanced offline functionality, especially in areas with poor connectivity.

Actions Taken

Enhanced Customization

Introduced more customizable expense categories, allowing users to tailor their tracking more effectively.

Improved Reporting Features

Added advanced filtering options for expense reports to enable users to analyze data by specific date ranges and categories.

Offline Functionality Upgrade

Optimized offline capabilities to ensure a more seamless experience when users are not connected to the internet.

Suggested Improvements for Future Updates

- **In-App Tutorials:** Implement guided tutorials to help new users navigate the app and utilize its features effectively.
- **Integration with Banking Apps:** Explore partnerships with banking apps for automatic expense tracking and synchronization.
- **Gamification Elements:** Introduce gamification features to encourage budget adherence and reward users for reaching financial goals.
- **Community Features**: Consider adding community forums or support groups within the app to foster user engagement and sharing of financial tips.
- **Regular Surveys for Feedback**: Establish a system for ongoing user feedback through regular surveys to continually adapt the app to user needs.

Q&A Preparation

1. What were the key design choices for the user interface?
Answer: We prioritized an intuitive user interface to ensure ease of navigation, especially for students who may not be tech-savvy. We implemented a clean layout with clear labels and simple icons, allowing users to log expenses quickly and access features without confusion. User feedback during the testing phase guided us in refining the design for better usability.
2. How did you ensure data security and privacy?
Answer: Data security was a top priority. We implemented encryption for user data storage and used hashing algorithms for password management, ensuring that sensitive information is protected. Additionally, we adhered to data privacy regulations and included features for users to manage their data preferences.
3. What challenges did you face during development?
Answer: One significant challenge was balancing the app's complexity with the team's limited technical experience. We had to prioritize core features that would deliver immediate value while planning for future enhancements. Additionally, ensuring offline functionality was complex but essential for our target audience, influencing many design decisions.
4. Why did you choose SQLite for data storage?
Answer: SQLite was chosen for its lightweight nature and ability to handle local data storage effectively. It allows the app to function smoothly offline, which is critical for our users. Its simplicity also aligns well with our team's technical expertise, making it a suitable choice for efficient data management.

Q&A Preparation

5. What lessons did you learn from user feedback?
Answer: User feedback highlighted the importance of customization and flexibility in expense tracking. We learned that providing users with options to tailor their experience enhances engagement and satisfaction. Regularly soliciting feedback will be crucial for ongoing improvements and ensuring the app remains relevant to user needs.
6. How do you plan to incorporate future updates?
Answer: We plan to implement a structured update schedule based on user feedback and market trends. Regular surveys will be conducted to gather insights on desired features and improvements. We also aim to stay agile in our development process, allowing us to respond quickly to user needs and technological advancements.
7. What features do you consider essential for the next version?
Answer: For the next version, we consider integrating in-app tutorials for new users, enhancing reporting features with more detailed analytics, and exploring partnerships for banking app integration. Additionally, gamification elements could motivate users to adhere to their budgets more effectively.

Conclusion

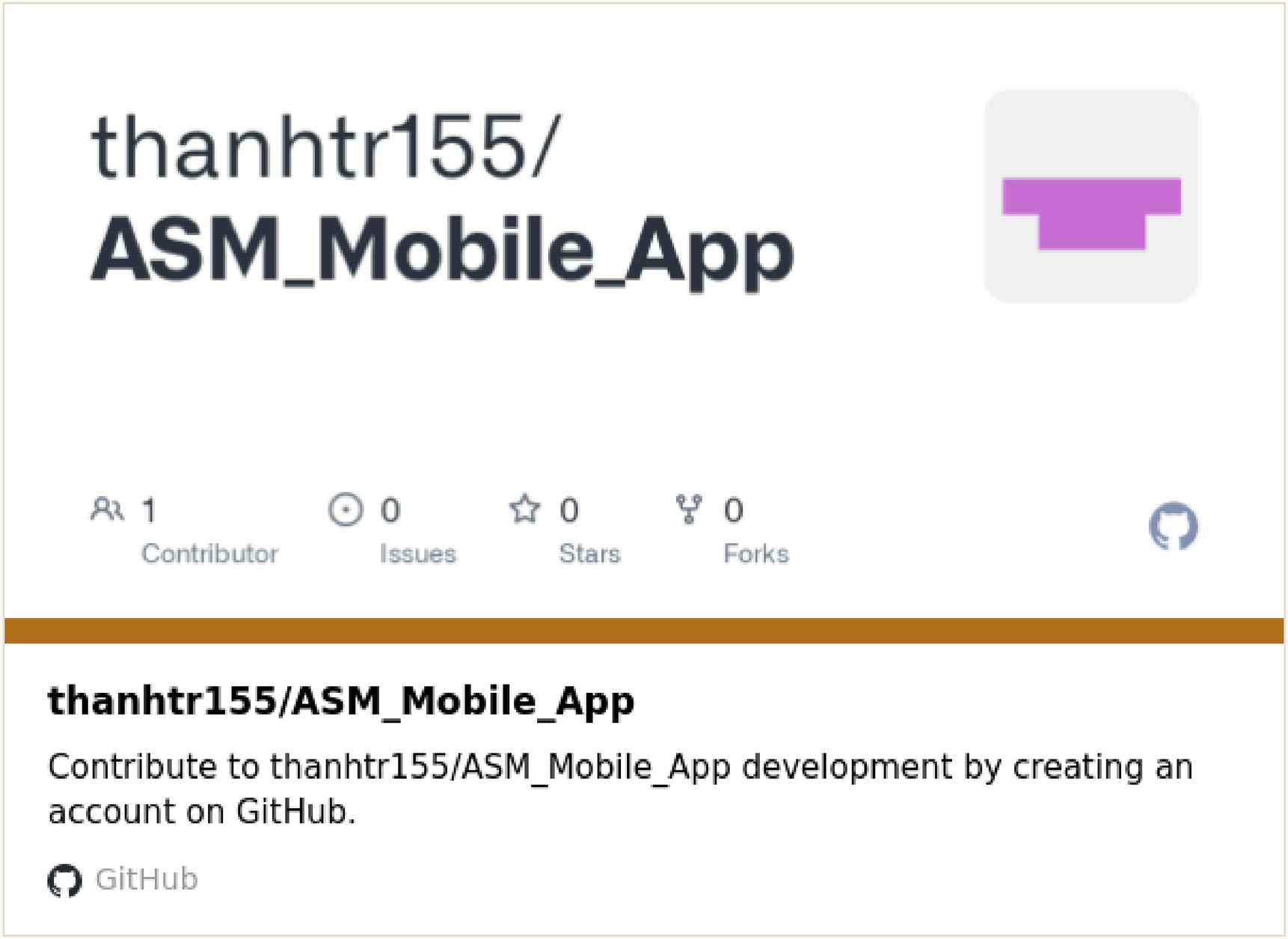
In conclusion, the CampusExpense Manager project has achieved significant milestones in developing a user-friendly financial management app tailored for university students. We successfully implemented core features such as expense tracking, budgeting alerts, and customizable categories, all designed to enhance students' financial literacy and management skills.

Reflecting on our team's efforts, we navigated various challenges, including technical constraints and time limitations, while fostering collaboration and adaptability. The commitment and creativity of each team member were instrumental in bringing this project to fruition.

We express our gratitude for the opportunity to present our work and share our insights. We look forward to the potential future enhancements and continued impact of the CampusExpense Manager app in helping students manage their finances effectively. Thank you!

Thank You

Link Github


thanhtr155/
ASM_Mobile_App
1
Contributor
0
Issues
0
Stars
0
Forks
thanhtr155/ASM_Mobile_App
Contribute to thanhtr155/ASM_Mobile_App development by creating an account on GitHub.
GitHub