CENTENNIAL COLLEGE

School of Engineering Technology and Applied Science

ICET Department

Mobile Application Development Project

MAPD-726

Project Plan: SNAPIFY

Submitted to: Prof. Vinayagathas Vaithilingam

By: Group 1

Name	Id
Bhargav Borse	301278352
Kajal Patel	301399333
Khanjan Dave	301307330
Rahul Edirinsinghe	301369991

1. Introduction

1.1. Overview

The Snapify E-commerce Platform is a dynamic project that aims to build a responsive and user-friendly e-commerce platform. Snapify Innovations Ltd. envisions a full solution built in Flutter using Agile techniques. The platform will have a User Dashboard for a streamlined shopping experience and an Admin Dashboard for effective products, focusing on the world of footwear for both men and women.

Key Highlights

1. **Fashion-Forward Shoe Collection:**
    - Dive into a world of footwear fashion as Snapify curates an extensive collection of shoes, catering to the diverse tastes and preferences of both men and women. From trendy sneakers to elegant heels, we offer a comprehensive range to suit every style.

2. **Flutter-Powered User Dashboard:**
    - Experience the convenience of shopping with our user-centric platform, designed and built using Flutter technology. Our User Dashboard ensures a seamless and visually appealing interface, providing an immersive journey for shoppers on both iOS and Android devices.

3. **Efficient Admin Dashboard:**
    - Behind the scenes, our Admin Dashboard empowers administrators to effectively manage the product catalog. Track inventory, analyze customer trends, and ensure the timely introduction of the latest shoe trends to the Snapify Shoe Emporium.

Vision

- Our vision encompasses embracing cutting-edge technologies, adopting best practices, and implementing Agile methodologies to ensure the Snapify Shoe Emporium platform is not just a destination for customers but a source of pride for our development team. We aim to craft a codebase that is scalable, maintainable, and a testament to our commitment to delivering top-tier software solutions.

- Also, our vision is to provide a seamless online shopping experience where customers can effortlessly discover, express, and elevate their style through our curated footwear selection. From casual chic to sophisticated elegance, every pair tells a unique fashion story, allowing customers to confidently stride through life in their chosen style.

1.2. Deliverables

The key deliverables for the Snapify E-commerce Platform are outlined below:

1. **User Dashboard Implementation:**
    - **Description:** Develop an intuitive and responsive user interface within the Snapify mobile app. Customers should be able to seamlessly browse, select, and purchase a diverse range of shoes.
2. **Admin Dashboard Development:**
    - **Description:** Create a robust administrative interface within the Snapify platform for efficient management of products, reports, and discounts.
3. **Database Integration:**
    - **Description:** Integrate a backend database to ensure secure and efficient storage and retrieval of product data, user information, and transaction details.
4. **Mobile App Deployment (iOS/Android):**
    - **Description:** Deploy the Snapify mobile application on both iOS and Android platforms, providing customers with a seamless and consistent user experience.
5. **Documentation Preparation:**
    - **Description:** Create comprehensive documentation, including user guides and technical specifications, to serve as a reference for users and developers.
6. **Security Testing:**
    - **Description:** Conduct thorough security testing to identify and address vulnerabilities, ensuring a secure online shopping environment.
7. **User Training Sessions:**
    - **Description:** Organize training sessions for users to familiarize them with the platform, enhancing their overall experience.
8. **Performance Optimization:**
    - **Description:** Optimize the performance of the Snapify platform to ensure quick load times and a smooth user experience.

1.3. Assumptions and constraints

Assumptions:

1. **Team Collaboration:**
    - *Assumption:* The project team will maintain open and effective communication throughout the development process, ensuring collaboration and timely problem resolution.
2. **Availability of Resources:**
    - *Assumption:* The necessary resources, including hardware, software, and skilled team members, will be consistently available to meet project requirements.
3. **Technology Stack Suitability:**
    - *Assumption:* The selected technology stack, including Flutter for mobile development, React for the API, and MongoDB for the database, is suitable for the project's requirements.

Working Timeline Assumptions:

The assumptions within the working timeline are based on the expectation that each sprint will progress smoothly, with team members fulfilling their assigned roles and responsibilities. It is assumed that the outlined goals for each sprint will be achievable within the designated time frames.

Constraints:

1. **Time Constraints:**
    - *Constraint:* The project must adhere to the specified timeline, with each sprint and milestone being completed within the allotted weeks.
2. **Security and Compliance:**
    - *Constraint:* The project must comply with all relevant security standards and legal regulations, potentially impacting the development process.
3. **External Dependencies:**
    - *Constraint:* External dependencies, such as third-party APIs or services, may impact project timelines and functionalities.

Working Timeline Constraints:

The working timeline constraints are centered around the fixed nature of the sprints, with specific deliverables and goals established for each. Any deviation from the planned schedule may impact subsequent sprints, potentially affecting the overall project timeline.

1.4. Risks and Assets

- Risks
  - Potential delays due to unforeseen technical challenges during Flutter development.
  - External dependencies, such as third-party APIs, may impact project timelines.
- Assets
  - The project team possesses significant expertise in Android, backend and Flutter development.
  - Access to a robust version control system and collaborative tools for efficient teamwork.

2. Management structure

2.1. Project life cycle

The Snapify E-commerce Platform project will use Agile approach. Agile is chosen because of its iterative and collaborative nature, which promotes flexibility, adaptability to changing needs, and constant input throughout the project life cycle. This method is consistent with the dynamic nature of software development, resulting in a responsive and customer-centric development process.

2.2. Project organization

Project lead:

- Name: Bhargav Borse
- Role: Business Analyst and Backend Developer
- Responsibility:
  - Analyze and document business processes, requirements, and objectives.
  - Use technologies such as Jira to handle requirements and track projects.
  - Develop precise functional specifications and user stories.
  - Develop and manage APIs for communication between the frontend and backend.
  - Improve server performance and scalability.
  - Collaborate with frontend developers to incorporate user-facing components.
  - Maintain data security and integrity in backend processes.

Team Members:

1. Name: Kajal Patel
   - Role: UI/UX Developer and Tester
   - Responsibility:
     - Design and develop user interfaces for web and mobile apps.
     - Create wireframes, prototypes, and final designs using programs such as Adobe XD, Sketch, or Figma.
     - Ensure that the UI is consistent with brand guidelines and responsive across several devices.
     - Carry out test plans to identify software faults, defects, and difficulties.
     - Create and implement test cases based on project requirements.
     - Collaborate with development teams to better understand features and functionality.

2. Name: Khanjan Dave
   - Role: Mobile App Developer
   - Responsibility:
     - Create mobile apps for Android and/or iOS platforms.
     - Write clean and efficient code.
     - Collaborate with UI/UX designers and backend developers to incorporate features.
     - Optimize applications' performance, usability, and security.
     - Stay informed on mobile development developments and best practices.

3. Name: Rahul Edirinsinghe
   - Role: Backend Developer
   - Responsibility:
     - Design and build server-side logic and databases.
     - Develop and manage APIs for communication between the frontend and backend.
     - Improve server performance and scalability.
     - Collaborate with frontend developers to incorporate user-facing components.
     - Maintain data security and integrity in backend processes.

2.3. Risk and Asset Management

1. Risk Management
   - Risks will be identified, evaluated, and managed continually throughout the project's life cycle.
   - Regular risk assessment meetings will be held to review and update the risk register.

- Mitigation methods will be implemented proactively in order to reduce the impact of potential risks.

2. Asset Management
   - Assets, such as the team's knowledge in Android, Flutter, and other technologies, will be used to improve project outcomes.
   - The continuous evaluation of team assets will influence decision-making and resource allocation.

2.4. Issue Management

- Issues will be handled using conventional Project Management methods.
- An issue log will be kept to document and track all discovered issues.
- The project team will work together to address and resolve difficulties as soon as possible so that they do not impede project development.

2.5. Communication

- Google Meet, Zoom, or MS Teams: Video conferencing tools for weekly meetings, updates, and collaborative discussions.
- GitHub: Primary platform for code collaboration, version control, and tracking changes.

3. Planning and control

3.1 Estimate

Task	**Estimated Staff Weeks**
Project Planning	1
Design	
- UI/UX Design	2
Implementation	
- Frontend Development	3
- Backend Development	3
Testing & QA	
- Test Planning & Preparation	0.5
- Test Execution & Bug Fixing	0.5

3.2 Resource Identification

For resource identification for an ecommerce application developed using Flutter for the frontend and Node.js for the backend

Flutter Developer: Responsible for developing the frontend of the application using Flutter.

Node.js Developer: Responsible for developing the backend of the application using Node.js.

UI/UX Designer: Responsible for designing the user interface and user experience of the application.

Business Analyst: Responsible for analyzing business requirements, gathering project needs, and coordinating between the development team and stakeholders.

We need to allocate resources accordingly. Let's assume a budget of $100,000 for the project over four months. Here's a rough breakdown of how we can allocate the budget among the team members:

Flutter Developer: $30,000 - $40,000
Node.js Developer: $30,000 - $40,000
UI/UX Designer: $20,000 - $30,000
Business Analyst: $10,000 - $20,000

3.3 Resource Allocation

Role	Allocation (Weeks)
Flutter Developer	4 weeks
Node.js Developer	4 weeks
UI/UX Designer	1 week
Business Analyst	1 week

This allocation ensures that each role has a proportionate amount of time within the 10-week project timeline.

3.4 Schedule

At the project's start, we focused on laying a strong foundation. We set up our project on GitHub and organized our tasks using Jira. In the following sprints, we worked on key features like user login, profiles, and product display, ensuring everything worked smoothly. Each step was carefully planned to keep us on track for future development.

Release 1

- **Sprint 1 (2/7/2024)**:
  - Initial project setup, including GitHub repository creation.
  - Setup of Jira for project management.
  - Implementation of splash, login, logout, and registration functionality on both frontend and backend, including comprehensive testing with administrative functionalities.

- **Sprint 2 (2/14/2024)**:
  - Development of user home screen on both frontend and backend.
  - Creation of user and admin profile screens.
  - Rigorous testing of implemented features.

- **Sprint 3 (2/21/2024)**:
  - Implementation of product categories and product detail pages on the frontend.
  - Thorough testing of frontend functionalities.

Release 2

- **Sprint 1 (3/6/2024)**:
  - Design and development of dashboard on the frontend.
  - Creation of product add screen.
  - Submission of completed tasks.

- **Sprint 2 (3/13/2024)**:
  - Implementation of cart screen and previous orders functionality.
  - Development of thank you page.
  - Completion of sprint tasks within the timeframe.

- **Sprint 3 (3/20/2024)**:
  - Creation of order processing screen for admin, enabling order status management.
  - Submission of tasks as per schedule.

Release 3

- **Sprint 1 (3/27/2024)**:
  - Integration of payment gateway and development of payment screen.
  - Testing and verification of payment functionalities.

- **Sprint 2 (4/3/2024)**:
  - Generation of comprehensive reports including category-wise and product sale reports, along with daily, monthly, and weekly sales analysis.
  - Submission of reports and completion of sprint tasks.

- **Sprint 3 (4/10/2024)**:
    - Final testing phase focusing on bug fixing and overall quality assurance.
    - Rectification of identified issues and ensuring the stability of the application.

Final Project Release and Documentation (4/17/2024)

- Completion and submission of all project deliverables.
- Documentation of the entire project lifecycle, including technical documentation, user guides, and any other necessary documentation for stakeholders.

This structured breakdown ensures clear communication of project milestones, deliverables, and timelines in a professional manner.

3.5 Tracking and Control

To effectively monitor and manage the project's cost and schedule, we will rely primarily on weekly status reports. These reports will provide a snapshot of the project's progress, detailing accomplishments, challenges, and any adjustments made to the schedule or budget.
In each weekly status report, we will include the following key elements:

1. **Project Milestones**: A summary of milestones achieved during the reporting period, highlighting progress towards project objectives.

2. **Task Updates**: An overview of tasks completed, in progress, or delayed, along with explanations for any deviations from the original schedule.

3. **Resource Utilization**: Tracking the utilization of resources, including manpower and budget, to ensure efficient allocation and prevent overruns.

4. **Budget Status**: A breakdown of budget expenditure for the week, comparing actual spending against the allocated budget.

5. **Schedule Updates**: Any adjustments made to the project schedule, including revised timelines for specific tasks or milestones.

In addition to weekly status reports, we will utilize other tools and methodologies specified in the course outline as necessary. This may include regular meetings with stakeholders to discuss project progress, budget reviews, and ad-hoc reports to address specific issues or concerns.
By implementing this streamlined approach, we aim to maintain clear visibility into the project's cost and schedule, identify potential issues early, and take proactive measures to keep the project on track.

4. Technical process

4.1. Environment

The development environment for the Snapify E-commerce Platform is established in Laboratory A3-68. This dedicated space provides a controlled and collaborative setting for the project team. Laboratory A3-68 is equipped with the necessary infrastructure, workstations, and resources to support the design, implementation, and testing phases of the project. It serves as a central hub for team discussions, code reviews, and collaborative problem-solving.

4.2. Methods, tools and technique

- The Agile methodology will be the guiding framework for the Snapify project. Embracing Agile principles ensures adaptability, collaboration, and customer-centric development. The iterative cycles of planning, execution, and feedback facilitate continuous improvement, allowing the team to respond effectively to changing requirements.
- Jira, a robust project management tool, will be instrumental in organizing and tracking tasks. Agile boards within Jira will be employed to plan sprints, visualize workflow, and manage project progress collaboratively. This tool enhances transparency and streamlines communication across the team.

4.3. Technology

Software Tools:

- **Flutter:**
  - *Purpose:* Utilized for mobile application development.
  - *Benefits:* Flutter, known for its expressive UI and fast development, will empower the team to create a visually appealing and responsive e-commerce application.
- **React:**
  - *Purpose:* Employed for API development.
  - *Benefits:* React's component-based architecture and efficient data handling make it an excellent choice for building a scalable and performant API that interacts seamlessly with the frontend.
- **MongoDB:**
  - *Purpose:* Selected as the backend database.
  - *Benefits:* MongoDB's flexible document-based structure and scalability will ensure efficient storage and retrieval of product data, contributing to a dynamic and responsive e-commerce platform.

Development Environment (IDEs):

- **Visual Studio Code (VSCode):**
  - *Purpose:* The primary integrated development environment.
  - *Benefits:* VSCode's lightweight yet powerful features, including extensions for Flutter and React development, will enhance the coding experience and productivity of the development team.

Version Control Tools:

- **Git:**
  - *Purpose:* Utilized for version control.
  - *Benefits:* Git's distributed version control system facilitates collaboration, code branching, and merging, ensuring a streamlined and organized development process.

Collaboration and Communication:

- **Google Meet, Zoom, or Microsoft Teams:**
  - *Purpose:* Selected for virtual communication.
  - *Benefits:* These platforms provide reliable channels for virtual meetings, updates, and collaborative discussions, fostering effective communication and coordination.

- **GitHub:**
  - *Purpose:* Centralized code repository hosting.
  - *Benefits:* GitHub offers collaborative features such as pull requests, issue tracking, and code documentation, facilitating efficient code management and review.

5. Supporting plans

5.1. Configuration Plans

Artifact Type	Description
Documentation	Comprehensive records of project details and procedures
Release and Iteration Plans	Plans detailing major releases and iteration schedules
User Stories	Narratives capturing end-user functionalities
Architecture Specs	Blueprints detailing the system's architectural design
Status Reports	Regular updates on project progress and milestones

Design Specs	Detailed specifications for the system's design
Source Code	The heart of our project, meticulously organized

- **Strategy:** Our Configuration Management revolves around a central repository, ensuring all project artifacts are stored, versioned, and accessible. Automated daily builds guarantee that the entire team is working with the latest configurations.
- **Implementation:**
  - **Repository Structure:** A well-organized structure for documents, plans, stories, specs, reports, and code.
  - **Version Control:** Each change will be versioned, providing a clear history of project evolution.
  - **Automated Builds:** Daily builds triggered by continuous integration tools to maintain synchronization.

5.2. Quality assurance

Quality assurance is an integral part of our development process, ensuring that both documentation and source code meet the highest standards. Our approach includes regular reviews and testing to guarantee the reliability and excellence of our deliverables.

Execution:

1. **Document Reviews:**
   - Periodic reviews will be conducted to ensure that all project documentation is accurate, clear, and aligns with the defined project goals.
   - Team members will collaboratively review documents, providing feedback and ensuring that information is up-to-date.
2. **Testing:**
   - Comprehensive testing will be conducted at various levels, including unit, integration, and system testing, to identify and rectify any issues in the source code.
   - Automated testing tools may be employed to streamline the testing process and enhance efficiency.

5.3. Testing

Our testing plan is exhaustive, covering all aspects of the system to ensure a flawless user experience.

Execution:

1. **Test Case Design:**
   - Our testing team will design meticulous test cases, mapping each case to specific project requirements.

- Test cases will be created to cover various scenarios, including positive and negative test cases.

2. **Testing Stages:**
    - Unit testing during the development phase will focus on individual components to ensure they function as intended.
    - Integration testing will examine the interactions between integrated components to identify any compatibility issues.
    - System testing, conducted before deployment, will validate the overall functionality and performance of the entire system.

5.4. Deployment

Platform: The Snapify E-commerce Platform will be deployed on Google Play for Android and the App Store for iOS.

Steps:

1. **Build Preparation:**
    - Before deployment, we will ensure that the final build is stable and free of critical issues.
    - A thorough testing phase, including regression testing, will be conducted to validate the stability of the build.
2. **Submission:**
    - Following platform-specific submission processes for Google Play and the App Store, adhering to their respective guidelines.
    - This involves preparing necessary metadata, screenshots, and other required assets for the app store listings.
3. **Accessibility:**
    - Ensuring the app is accessible to a wide audience by adhering to industry standards and guidelines such as WCAG (Web Content Accessibility Guidelines).
    - This includes making sure the app is compatible with assistive technologies and provides an inclusive experience for users with disabilities.