Software Project Management Plan

TastyGO

Table of Contents:

1. Overview

1.1 Project Summary

1.1.1 Project Purpose, Goals and Objectives

Project Purpose	
The TastyGO application aims to provide an online food-ordering and management platform for customers, restaurant clients, and administrators. Its purpose is to streamline ordering, menu management, payment, and customer service.	
Project Goal	**Project Objectives**
Goal 1: Create an intuitive, user-friendly interface for customers, restaurant clients, and admins.	**Ensure clear navigation** with minimal clicks or confusion across all user roles.
	Maintain a consistent look and feel (color schemes, button styles, fonts) across the application.
	Implement responsive design so that the platform is usable on desktops, tablets, and smartphones.
Goal 2: Provide reliable order management, payment processing, and menu updates.	**Integrate a secure payment gateway** (credit card, wallet, etc.) with detailed transaction logs.
	Enable real-time menu updating so restaurants can add/edit/remove items seamlessly.
	Establish concurrency controls to handle multiple simultaneous user orders without data conflicts.

Goal 3: Offer real-time insights and analytics for restaurants and administrators.	**Generate sales and order-tracking reports** (daily, weekly, monthly) with user-friendly visualizations
	Include customer feedback metrics for restaurants to see ratings and reviews in one place.
	Provide an admin dashboard summarizing platform-wide activities, user accounts, and potential issues.

1.1.2 Assumptions and Constraints

- Team members will attend all meetings
- Team members will meet all the deadlines
- Team members will follow the requirements specified in SRS
- The application is being designed to only run on Java supported Platforms
- Team members will work on the project outside the class to finish it on time

1.1.3 Project Deliverables

This project will deliver the following items

- Working Executable Application
- Requirements Document, SPMP, SQAP, Object-Oriented Analysis/Design Documents, Test Plan, Documented Source Code, Maintenance Manual

1.1.4 Schedule and Budget Summaries

There is **no budget** allocated for this course project. The timeline is based on course deliverables:

- **Requirements** (Due: January 31, 2025)
- **Software Project Management Plan** (SPMP) (Due: February 7, 2025)
- **Object-Oriented Analysis** (Due: March 7, 2025)
- **Object-Oriented Design** (Due: March 21, 2025)
- **Final Submission** (Due: April 4, 2025)

1.1.5 Project Success Criteria

Category	Success Metric	Measurement Method	Target	Assigned to

Scope	All core features (menus, ordering, payment, admin console)	Requirements Document vs. Implementation	100% coverage of required functionality	Project Managers
Schedule	Submission deadlines (SPMP, OOA, OOD, Final) met	Milestone reviews	100% on-time deliverables	Everyone
Quality	Low defect density & user-friendly interface	QA metrics, user testing feedback	< 5% user-reported defects post-release	QA Tester (Desman)
Stakeholder satisfaction	Positive feedback from client (instructor) & participants	Review sessions, acceptance testing	Client sign-off; high acceptance rating	Customer Feedback Coordinators
Risk management	Identified risks mitigated or resolved in time	Risk log updates, negative impact events on schedule	< 10% schedule/budget overruns due to risks	Project Managers

1.2 Project Governance

Area	Description
Instructor (Client)	Approves changes to scope, receives deliverables.
Project Managers (Samir, Rusanth)	Coordinate day-to-day decisions, schedule, and changes.
Team Members	Execute tasks according to roles.

1.3 Evolution of the Plan

The management team will meet with the client on Monday, Feb 10, discussing application expectations and functionality. Keeping in mind the final project deadlines and the client's feedback, we will iterate on requirements, UI/UX design, and scope. Future versions of this plan may evolve if the instructor requests changes in scope or schedule.

2. References

- IEEE Template

https://mobileappdev.academic.csusb.edu/wp-content/uploads/2 019/0

- SPMP_Standard_IEEE_1058

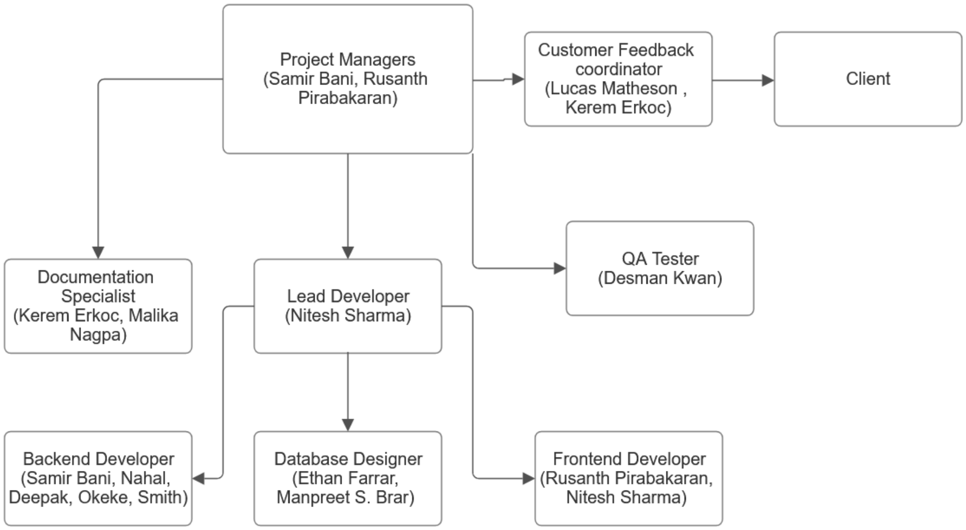
https://mylearningspace.wlu.ca/d2l/le/content/569520/viewContent/4028192/View

3. Definitions

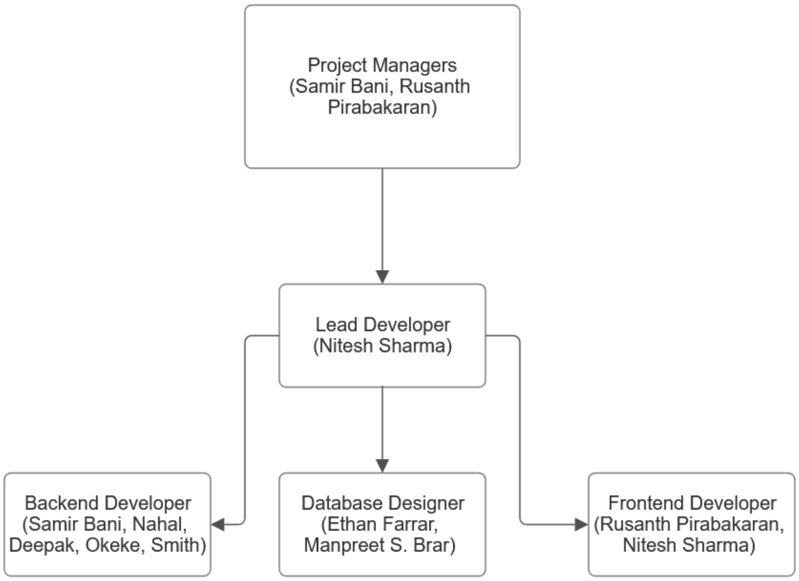
- UI: User Interface - means by which a user and system interact
- GUI: Graphical User Interface - Interface represented by buttons and visuals.
- SRS - Requirements Document

4. Project Organization

4.1 External Interfaces


Project Managers
(Samir Bani, Rusanth Pirabakaran)
Customer Feedback coordinator
(Lucas Matheson , Kerem Erkoc)
Client
QA Tester
(Desman Kwan)
Documentation Specialist
(Kerem Erkoc, Malika Nagpa)
Lead Developer
(Nitesh Sharma)
Backend Developer
(Samir Bani, Nahal, Deepak, Okeke, Smith)
Database Designer
(Ethan Farrar, Manpreet S. Brar)
Frontend Developer
(Rusanth Pirabakaran, Nitesh Sharma)

4.2 Internal Structure


Project Managers (Samir Bani, Rusanth Pirabakaran)
Lead Developer (Nitesh Sharma)
Backend Developer (Samir Bani, Nahal, Deepak, Okeke, Smith)
Database Designer (Ethan Farrar, Manpreet S. Brar)
Frontend Developer (Rusanth Pirabakaran, Nitesh Sharma)

4.3 Roles & Responsibilities

Names	Role	Responsibilities
Samir Bani	Project Manager, Backend Dev	Oversees schedule & deliverables, implements backend modules.
Rusanth Pirabakaran	Project Manager, Frontend Dev	Oversees schedule & deliverables, implements frontend interfaces.
Nitesh Sharma	Lead Developer, Frontend Dev	Maintains coding standards, ensures code efficiency, and performs code reviews.
Ethan Farrar & Manpreet Brar	Database Designers	Design and maintain database schema, ensure data integrity.
Desman Kwan	QA Tester	Leads module & integration testing, enforces quality metrics.

Kerem Erkoc & Malika Nagpal	Documentation Specialists	Prepare SPMP, design docs, and any user documentation.
Lucas Matheson	Customer Feedback Coordinator	Liaises with client/stakeholders, communicates project progress and changes.

5. Managerial Process Plans

5.1 Start-up Plan

5.1.1 Estimation Plan

- Discuss with the client all the specifications for the application.
- Research all the technologies required to meet those expectations
- Figure out how much could be accomplished in the given time
- Create estimated deadlines and task completion deadlines to deliver the finished product on time

5.1.2 Staffing Plan

The project staff was selected by the professor, One member has left since the original group creation.

5.1.3 Resource Acquisition Plan

All the software required for this project is free. All group members will be using personal devices to work on the project with free software.

5.1.4 Project Staff Training Plan

All the group members have experience in the chosen software and programming language. Additional training will be done by the individual member to be functional for the project.

5.2 Work Plan

5.2.1 Work activities Work Units:

- **GUI Prototype**: The first prototype will be the basic non-functional GUI design with a description of how it will work.
- **Object-Oriented Design**: This will include the flow of classes, objects,

and entities interacting with each other, the function descriptions, and function-based use cases.

- **Final Delivery**: The final fully functional project with minimal faults.
- **Documentation**: All members will make sure that all the code that they write is properly documented.

5.2.2 Schedule Allocation

January 11, 2025	January 31, 2025	Requirements
January 31, 2025	February 7, 2025	SPMP
February 7, 2025	March 7, 2025	OO Analysis
March 7, 2025	March 21, 2025	OO Design
March 21, 2025	April 4, 2025	Final Submission

5.2.3 Resource Allocation

Each member has access to the same resources as all the other members. The resources include coding IDEs, personal devices, and other softwares.

5.2.4 Budget Allocation

No budget has been allocated for this project.

5.3 Control Plan

5.3.1 Requirements Control Plan

Each member of the group is required to attend the meetings whether in person or virtually. Also each member is required to document their code, follow the guidelines decided, and meet each deadline as well. Any unexpected issues, technical difficulties or requests by the clients will be assessed by the Managing team and decided upon.

5.3.2 Schedule Control Plan

Managers will make sure that each member is completing their tasks on time. The managing team will keep a constant eye on the progress of the project and make sure that everything is completed by the deadline. The Project Managers will keep all group members up to date on the progress of the project.

5.3.3 Budget Control Plan

There is no budget allocated to the project.

5.3.4 Quality Control Plan

The quality assurance and development team will periodically perform a quality check on the software to make sure that the project meets all the expectations. Also, the client will be present during the development of the application and therefore will be informed of the quality of the application.

5.3.5 Reporting Plan

The customer feedback coordinator will notify the client of the progress and the milestones. The customer feedback coordinator will discuss any changes and plans with the client in any planned meetings.

5.3.6 Metrics Collection Plan

The lead developer will make sure that at every milestone each developer completes their tasks so that the project is on track and progressing as expected. The lead developer will also make sure that the code is efficient, and meets all the requirements.

5.4 Risk Management Plan

Since there is no existing product for comparison, we won't be able to run the new system in parallel with another. As a result, extensive testing will be essential to ensure functionality and reliability. Frequent deadlines, regular meeting, and communication will be essential to make this project successful

Assuming that the client has limited experience with computers, we will focus on making the product as user-friendly as possible. Clear communication and a well-structured workflow will be key to ensuring the client can effectively use the system.

To minimize the risk of major design flaws, rigorous testing will be implemented into the development process. Each team member will first test their own code before reviewing another team member’s work. Our Q&A will oversee integration testing and handle overall product testing.

Although the risk of hardware failure is low, contingency plans are in place—if a machine fails, a replacement will be used.

5.5 Close Out Plan

- All code merged into the GitHub repository with final documentation.
- Presentation/demonstration on the final day.
- Final sign-off by instructor.

6. Technical Plan Process

6.1 Process Model

We are using the **Unified Process** with iterative increments.

6.2 Methods, Tools, and Techniques

- **Method**: Unified Process, iterative approach.
- **Tools**: Java, HTML
  - **Development**: Java, HTML, CSS, JavaScript (as needed for front-end), a relational DB for data storage.
  - **Version Control**: GitHub (with branching and pull requests).
- **Techniques**:
  - **Frequent Meetings**: Weekly stand-ups to sync tasks.
  - **Code Reviews**: Each pull request is reviewed by at least one other developer.
  - **Prototype Demonstrations**: Early demonstration of UI flows for feedback from the client/instructor.

6.3 Infrastructure Plan

- Primarily local development; no special servers.
- GitHub repository for continuous integration and version control.

6.4 Product Acceptance Plan

- Instructor is involved in progress reviews.
- QA verifies features meet Requirements Document.
- Final acceptance sign-off after integration testing.

7. Supporting Process Plans

7.1 Configuration Management Plan

We use **Git** for all artifacts, branching for major features, code review before merging.

7.2 Verification and Validation Plan

- **Module/Unit Testing:**
  - Use **JUnit** (or some testing framework) for Java-based logic.
  - Each developer writes tests for their own modules before code is merged.

- **Integration & System Testing:**
  - QA tester (Desman) coordinates multi-module test cases to ensure end-to-end functionality.
- **Acceptance Testing:**
  - Instructor reviews partial deliverables against the Requirements Document.
  - Bug reports documented in GitHub issues or a shared spreadsheet.

7.3 Documentation Plan

- Managers (Samir, Rusanth) coordinate SPMP.
- Documentation Specialists (Kerem, Malika) maintain design docs, user guides.
- Code-level documentation is developer responsibility.

7.4 Quality Assurance Plan

- **Standards & Guidelines:**
  - Unified coding conventions (naming, indentation, comments).
  - Documentation Specialists (Kerem, Malika) ensure final textual consistency in all artifacts.
- **Reviews:**
  - Design Reviews before major coding tasks.
  - Code Reviews on each merge request to maintain code quality.
- **Testing Tools:**
  - **JUnit** for unit tests, potentially Selenium for UI tests, manual test scripts for complex scenarios.
- **Metrics:**
  - Track fix turnaround time - set up a checklist.

7.5 Reviews and Audits

During development and testing phases, every member of the development team will test and report any deficiencies in the mobile app. Design flaws or bugs will be reported and documented for immediate fix or future review.

7.6 Problem Resolution Plan

- Team escalates issues to Project Managers.
- If unresolved, the instructor is consulted for guidance.

7.7 Subcontractor Management Plan

N/A (No subcontractors)

7.8 Process Improvement Plan

- Document best practices during each phase.
- Defer non-critical features to post-release if time constraints require.

8. Team Member Participation & Rating

Team Member Participation Record

- Each member tracks hours/contributions (e.g., feature commits, documentation updates).
- Project Managers compile these records bi-weekly.

Team Member Rating

- At the end of the project, each member assigns a rating from 0 to 5 for fellow members' contributions (5 = expected contribution, 0 = no contribution).
- The entire team must review and approve the rating document.

This rating process will be documented and submitted with the final updated SPMP.

9. Software Quality Assurance (SQA) Plan

9.1 Quality Objectives

- We will ensure by testing that the software meets all the requirements both functional and non-functional as specified in the project documentation.
- Aim and achieve a defect density of less than 1 per 1000 lines of code upon release.
- Ensure that at least 95% of the test cases are successful before project delivery.
- Our team will conduct regular code reviews and automated testing to maintain code quality.
- Our team will maintain comprehensive documentation for software specifications, testing procedures and error handling.
- We will also implement continuous integration (CI) and version control to facilitate smooth collaboration and development.

9.2 Roles And Responsibilities

Role	Name	Responsibilities
QA Tester	Desman Kwan	Leads testing efforts, verifies software compliance, ensures defect tracking and resolution

Project Manager	Samir Bani, Rusanth Pirabakaran	Ensures adherence to the quality standards, enforces testing schedules and oversees reviews.
Developers	Team Members	Write unit tests, document codes properly ensures code quality to best practices
Documentation Specialist	Kerem Erkoc, Malika Nagpal	Maintain updated quality documentation and test reports.
Customer Feedback Coordinator	Lucas Matheson	Collect feedback and reports from the customers about usability issues.

9.3 Standards & Procedures

This section ensures alignment with the Quality Assurance Plan (7.4) and Verification & Validation Plan (7.2) to maintain consistency across all processes:

- Coding Standards are defined under 7.4 Quality Assurance Plan, which developers must adhere to.
- Testing Procedures follow the methodologies outlined in 7.2 Verification & Validation Plan, including unit testing, integration testing, and user acceptance testing.
- Bug Tracking will be conducted through GitHub, and all critical defects will be resolved before release.
- All testing and compliance activities will align with IEEE Standard 1058, 1998.

9.4 Verification and Validation Activities

The Verification & Validation Plan (7.2) details the testing approach, including unit testing, integration testing, and acceptance testing. This section will reference those activities and highlight additional checks:

- **Static Code Analysis** will be performed using tools such as SonarQube etc.
- **Regression Testing** will be conducted before each milestone to prevent breaking changes.

- **Performance Testing** will verify system responsiveness under high user load.
- **Security Testing**: Validates protection against unauthorized access and data breaches.

9.5 Documentation

While general documentation is covered in 7.3 Documentation Plan, the following artifacts will be maintained as part of the SQA process:

- **Test Plan**: Describes testing scope, methodologies, and success criteria.
- **Defect Reports**: Track issue status and resolution progress.
- **Test Case Logs**: Records of executed tests and outcomes.
- **Final Validation Report**: Summarizes test results before release.

9.6 Continuous Improvement

Continuous improvement efforts will focus on ongoing quality assessments:

- Conducting **post-mortem analysis** after each major milestone.
- Using **automated monitoring tools** to detect performance degradation.
- Incorporating feedback from **customer feedback coordinator** for usability improvements.
- Adjusting testing methodologies based on defect trends.

9.7 Independence of SQA Group

To ensure unbiased testing, the SQA team operates independently from the development team:

- QA Tester (Desman Kwan) oversees validation and has final authority on defect resolution.
- Code cannot be merged into the main branch without passing predefined test cases.
- Stakeholder feedback collected by the Customer Feedback Coordinator is used for final quality evaluation before release.
- Developers and testers have separate responsibilities to prevent conflicts of interest.

By implementing this **Software Quality Assurance Plan**, we ensure that the **TastyGO application** meets all required standards and delivers a reliable, user-friendly experience.