Review and Recommendation System For ShopSmart (RRSS)
Project Plan

1 Introduction

The following Project Plan Document outlines the strategic framework and execution roadmap for the development of an online review and recommendation system for ShopSmart, titled as 'RRSS'. In the contemporary landscape of 21st century digital commerce, where consumer preferences and feedback hold substantial influence over purchasing behaviors, the Review and Recommendation System for ShopSmart (RRSS) endeavors to furnish users with invaluable insights sourced from peer reviews, coupled with bespoke product recommendations. This initiative is geared towards elevating the overall shopping journey for users, harnessing the synergy of user-generated content and cutting-edge algorithms to curate tailored product suggestions. This document includes several sections such as 'Project Organization', 'Development Process & Measurements', 'Project Milestones & Objectives', 'Deployment' and 'Lessons Learned' to ensure successful delivery of the RRSS within the anticipated timeline and budget constraints.

2 Project organization

Team Member	Software Developer	Software Architect	Project Manager	Software Tester	Software Analyst	Configuration Manager
Mert Tazeoğlu	X					X
Tarık Sümer	X	X				
Asım Ateş	X		X			
Kenan Gökdeniz Acet	X			X		
Emre Can Şahin	X				X	

2.1.1 - Software Developer (@ Everyone)

A software developer is a professional who specializes in crafting, building, implementing, and maintaining software applications or systems. Their main responsibility lies in transforming user needs into functional software solutions, employing programming languages, frameworks, and development tools. Throughout the software development lifecycle, developers are involved in various stages including analysis, design, coding, testing, deployment, and ongoing maintenance. Main responsibilities of software developers can be summarized as below:

- Researching, designing, implementing and managing software programs
- Identifying areas for modification in existing programs and subsequently developing modifications
- Deploying software tools, processes and metrics
- Developing quality assurance procedures
- Writing and implementing efficient code
- Testing and evaluating new programs; maintaining
- Determining operational practicality

• Upgrading existing systems

2.1.2 - Responsibilities of Software Architect (@ Tarık Sümer)

A software architect is a professional tasked with designing and overseeing the structure of a software system or application. Their primary responsibility involves making key design decisions, establishing technical standards, and ensuring that the software's overall architecture meets specific quality criteria like performance, scalability, reliability, and security. Main responsibilities of software architects can be summarized as below:

• Understanding the requirements and constraints
• Choosing and evaluating technology
• Understanding that how the architecture works
• Involvement in the hands-on elements of software delivery (designing and coding)
• Ownership of the architecture throughout the delivery
• Introduction and adherence to standards and principles
• Guidance and assistance to team members

2.1.3 - Responsibilities of Project Manager (@ Asım Ateş)

A project manager is a professional tasked with supervising the planning, execution, and finalization of a project within defined parameters such as time, budget, and scope. Their primary duty involves coordinating resources, tasks, and efforts to ensure that project objectives are achieved efficiently. They serve as a central communication hub, in order to facilitate collaboration among team members, stakeholders, and other project participants. Main responsibilities of project managers can be summarized as below:

• Activity and resource planning
• Organizing and motivating a project team
• Controlling time management
• Cost estimating and developing the budget
• Ensuring customer satisfaction
• Analyzing and managing project risk
• Monitoring progress
• Managing reports and necessary documentation

2.1.4 - Responsibilities of Software Tester (@ Kenan Gökdeniz Acet)

A software tester is a professional tasked with examining software applications or systems to uncover defects, bugs, and flaws prior to their release to end-users. Their core responsibility involves creating and executing test cases, analyzing test outcomes, and providing feedback to developers to guarantee that the software meets quality benchmarks and user needs. Main responsibilities of software testers can be summarized as below:

• Reviewing software requirements and preparing test scenarios
• Executing tests on software usability
• Analyzing test results on database impacts, errors or bugs, and usability
• Preparing reports on all aspects related to the software testing carried out and reporting to the software designers
• Participating in design reviews and providing input on requirements, product design, and potential problems

2.1.5 - Responsibilities of Software Analyst (@ Emre Can Şahin)

A software analyst, also referred to as a systems analyst or business analyst, is a professional specializing in the examination and assessment of software systems, processes, and needs to determine how technology can address business challenges or enhance operational efficiency. Main responsibilities of software analysts can be summarized as below:

• Gathering and analyzing user requirements, business objectives, and system functionalities
• Clarifying project goals, defining project scope, and setting objectives for development projects
• Writing detailed specifications, user stories, use cases, and functional requirements to meet needs
• Analyzing existing software systems and business processes to identify areas for improvement
• Assisting with system design, testing, deployment, and user training to ensure successful delivery
• Tracking project milestones, identifying risks, and addressing issues to keep projects on schedule

2.1.6 - Responsibilities of Configuration Manager (@ Mert Tazeoğlu)

A configuration manager is a professional who oversees the configuration management process within an organization or project. Their main responsibility is to ensure that configuration items (CIs) are properly controlled, tracked, and documented throughout the software development lifecycle. Main responsibilities of configuration managers can be summarized as below:

• Developing and implementing policies, standards, and guidelines for configuration management
• Defining and documenting the various elements that make up the software or system
• Managing different versions of configuration items to track changes accurately
• Reviewing & approving proposed changes, assessing their impact, and coordinating implementation
• Coordinating the release and deployment of software and system configurations
• Conducting regular audits to ensure that the actual configuration matches the documented baselines

2.2 - Neighboring / Referencing Projects

In E-Commerce area, there are numerous sites for neighbouring or referencing. Some popular local and worldwide examples of systems that enables customers to submit reviews and ratings for products, alongside offering tailored product suggestions based on individual preferences are below:
• **Amazon:** https://www.amazon.com.tr/
• **eBay:** https://www.ebay.com/
• **Trendyol:** https://www.trendyol.com/
• **Hepsiburada:** https://www.hepsiburada.com/
• **N11:** https://www.n11.com/

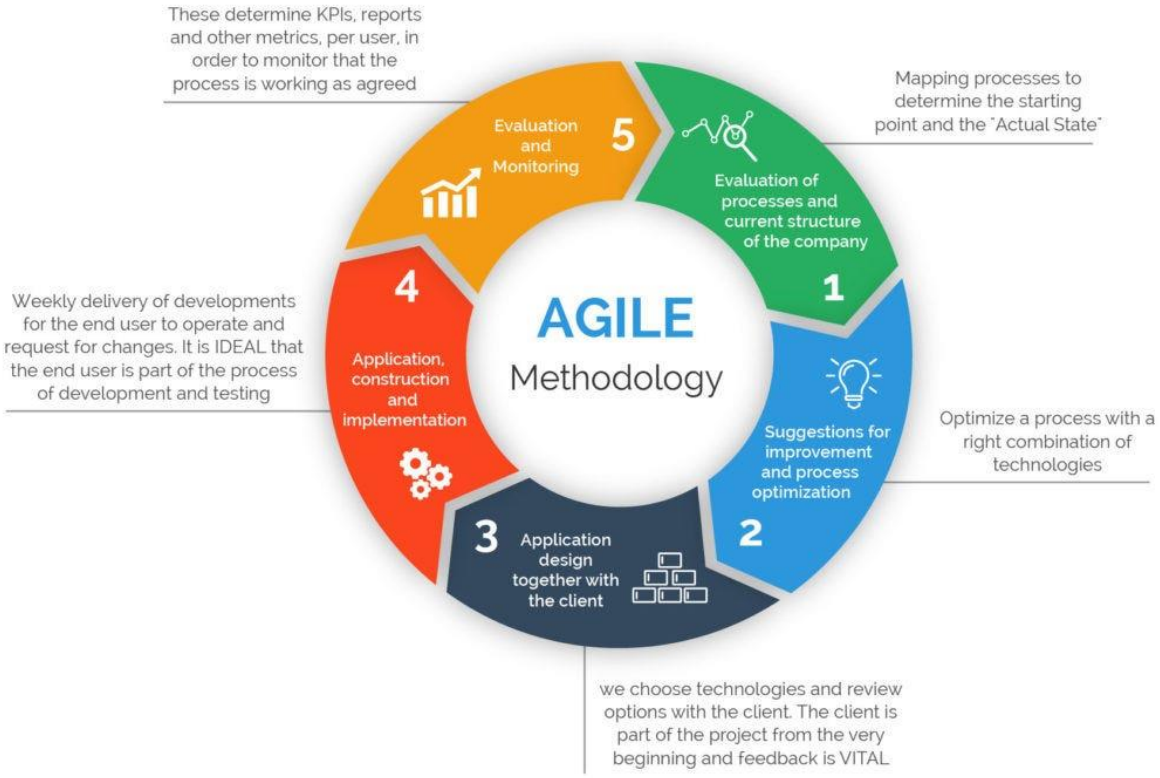
Furthermore, various popular applications offer features allowing users to provide product evaluations and receive personalized product recommendations across different sectors, even if transactions aren't facilitated within these platforms. Some prominent examples include:
• **Google Maps:** Google Maps incorporates user evaluations and ratings for various businesses, delivering location-specific recommendations. (Link: https://www.google.com/maps)
• **TripAdvisor:** Tripadvisor furnishes users with reviews and suggestions concerning travel destinations and lodging options. (Link: https://www.tripadvisor.com/)
• **Goodreads:** Goodreads serves as a platform where users can share reviews, ratings, and recommendations regarding books. (Link: https://www.goodreads.com/)

3 Development Process and Measurements

3.1 - Development Process With Agile Approach

Choosing the Agile methodology for this project's development is grounded in its demonstrated adaptability, efficiency, and focus on customer needs. Agile frameworks offer a structure that supports change, fosters ongoing enhancement, and enables the quick delivery of premium products. Here are the key motivations behind selecting Agile for our project management approach:

• **Adaptability and Flexibility:** Agile sets itself apart from traditional waterfall methods by allowing continuous reassessment and adjustment of plans. This adaptability is essential for keeping pace with shifts in market trends, technological advancements, and evolving project objectives, ensuring the final product stays relevant and valuable.
• **Progressive Delivery:** Agile advocates for the progressive release of the product in small, manageable portions, or sprints. This approach allows for the periodic release of product segments, facilitating early feedback and necessary modifications.
• **Improved Team Collaboration and Communication:** Agile places a strong emphasis on direct communication and close teamwork among project participants, including developers, project managers, and clients. This environment of cooperation enhances team commitment and project ownership.
• **Effective Risk Management:** Agile methods facilitate early risk identification and intervention. Regular iterations enable quick detection of potential issues, allowing the team to address risks promptly. This forward-looking approach to risk management diminishes the chance of project setbacks.
• **Continuous Quality Enhancement:** The iterative essence of Agile permits ongoing testing, assessment, and improvement of the product. This constant quality control process guarantees the development of a superior product, with flaws being detected and corrected quickly.



3.2 - Tracking Development Process and Measurements

Tasks	**Date Range**	**Work Load** (Weeks)	**Weekly Hour**
Software Project Managements	01.03.2024 - 24.05.2024	12 Weeks	5-7 Hours (avg.)
Software Project Vision and Plan	01.03.2024 - 15.03.2024	2 Weeks	4 Hours
Software Project Requirements	15.03.2024 - 29.03.2024	2 Weeks	6 Hours
Software Project Design	22.03.2024 - 19.04.2024	4 Weeks	8 Hours
Software Project Development	05.04.2024 - 24.05.2024	7 Weeks	12 Hours
Software Project Test	13.04.2024 - 24.05.2024	6 Weeks	5 Hours

4 Project Milestones and Objectives

Phase	**Iteration**	**Primary objectives** (risks and use case scenarios)	**Scheduled start or milestone**	**Target velocity**
		Software Vision & Project Plan		

Initiation	I1	**Documents Risk:** The project may face risks due to inaccurate planning of the project, inadequate time management, inexperience and lack of team members. **Mitigate Risk:** The existing team members need to have clear divisions of labor assigned to them. This can be achieved by referring to similar projects or seeking guidance from experienced professionals in the field.	01.03.2024 - 15.03.2024	8 Hours
Planning	I2	**Software Requirements Document Risk:** The software tools selected may be inadequate or improperly chosen for the project's needs. Additionally, there might be a shortage of labor due to difficulties in adapting quickly enough. **Mitigate Risk:** A research through to identify the necessary tools will be conducted. Gradually completing the documentation work until the deadline should help us avoid any difficulties.	15.03.2024 - 29.03.2024	14 Hours
Design & Development	I3	**Architectural Notebook & List of System Test Case Definitions Risk:** The project may face risks due to difficulty for meeting all requirements due to coinciding with exam week, as well as challenges stemming from inexperience and communication issues within the software development team. **Mitigate Risk:** Taking into account these challenges, it's advisable to start exam preparations early. Furthermore, ensuring constant communication among software developers through daily progress reports is essential.	22.03.2024 - 19.04.2024	30 Hours
Development & Monitoring &	I4	**Software Design Document (UML Model) & Coding Standard Risk:** Mistakes in the UML model can result in significant errors in the code. Issues identified later in the development phase can result in wasted time and effort. **Mitigate Risk:** Understanding the project	19.04.2024 - 10.05.2024	40 Hours

Testing		thoroughly and carefully drawing the UML diagram is crucial. Prior to initiating the development process, version differences and conflicts must be resolved to prevent future issues.		
Development & Verification & Validation & Testing & Closure	I5	**Software Test Result Report & Final Product Risk:** If the development time exceeds our initial estimates, the execution of our test cases may become compromised. The test scenarios, formulated by someone familiar with the project, may prove insufficient. There may be insufficient time to address issues in our tests before the final product is released. **Mitigate Risk:** Each step of development must be tested to avoid encountering surprises in the future. All stakeholders should be considered, and test cases should be written for each of them.	17.05.2024 - 24.05.2024	20 Hours

Note that, in the Agile approach the phases and timelines of a project are typically interwoven and flexibly planned. The Agile methodology supports an open and iterative development process rather than progressing according to a strict timetable. This reflects Agile's dynamic nature and flexibility in the project development process.

5 Deployment

Our software deployment strategy emphasizes simplicity and team collaboration. We will use GitHub for version control, allowing each team member to work on separate branches for safe experimentation and development of new features. This approach enables us to maintain a clear history of changes and facilitates easy collaboration.

When a feature is ready, the developer will create a pull request. This step involves a peer review process to ensure code quality and adherence to project standards. Once approved, changes will be merged into the main branch, marking the feature as ready for deployment.

For deploying updates to the production environment, we will adopt a manual deployment strategy to maintain control and ensure stability. This process will be periodically reviewed to incorporate feedback and make adjustments as necessary, aiming for a balance between speed and reliability in our software updates.

6 Lessons learned

• Thorough planning and a detailed analysis of project requirements from the beginning are vital. This approach establishes a clear path forward and aids in early identification of potential obstacles. The

importance of diligent planning and careful management of the project timeline cannot be overstated for the project's overall success. In determining the project's velocity, an emphasis is placed on flexibility, allowing for adjustments to unexpected events, thereby guaranteeing the project finishes on schedule without any setbacks.

• Effective time management is pivotal for project success. Establishing realistic timelines and adhering to them prevents last-minute rushes and ensures high-quality deliverables. For the same reason, proactively identifying potential risks and developing mitigation strategies is essential.

• Clear division of tasks among team members, according to their strengths and expertise, significantly enhances project's efficiency.

• Thorough testing prioritized at every stage of development ensures the early detection of issues and facilitated the delivery of a more robust final product.

• Thorough documentation throughout the project lifecycle is invaluable. It not only facilitates smoother onboarding and knowledge transfer but also serves as a reference point for resolving disputes and guiding future projects.

• Larger projects introduce more complexity, including increased coordination efforts and higher risk of scope creep. Regular team meetings and maintaining a dynamic project management approach is crucial in navigating these challenges.