Revisions			
Version	**Description**	**Date**	**Person**
1.0	The document was created.	04.11.2023	Elif Beril Sayli Betül Ergin Özde Uysal Annie Yang
1.1	The document was revised according to Iteration 1 feedback.	11.11.2023	Elif Beril Sayli Betül Ergin Özde Uysal Annie Yang
1.2	The document was updated.	25.11.2023	Elif Beril Sayli Özde Uysal Annie Yang
1.3	The document was revised according to Iteration 2 feedback.	02.12.2023	Elif Beril Sayli Özde Uysal Annie Yang
1.4	The document was updated for Iteration 3.	16.12.2023	Elif Beril Sayli Özde Uysal Annie Yang

Samwise Service App

Project Plan

1. Introduction

The project plan for the Samwise Service App provides a comprehensive roadmap for the development of our web-based service platform. It outlines the key components, practices, and measurements that will guide the project to success. The plan encompasses the project organization, practices, and metrics used, along with milestones and objectives for each iteration. The project is built on an iterative development approach, emphasizing continuous improvement and the delivery of incremental updates. This plan also addresses the deployment strategy, support and maintenance, security measures, and user communication. Finally, it addresses lessons learned, which will be gathered during retrospectives at the end of each iteration, ensuring the project's adaptability and continuous enhancement.

2. Project Organization

2.1. Project Team Roles and Responsibilities

The project team consists of three members: Elif, Annie, and Özde. Here are the roles and responsibilities of the team members:

Annie (Project Manager and Software Architect):
Responsibilities:

- Overall project planning, execution, and delivery within the defined scope, timeline, and budget.
- Communication and collaboration with stakeholders, ensuring their requirements are understood and

addressed.
- Risk management and mitigation planning.
- Team coordination and task assignment, ensuring each team member's workload aligns with project goals.
- Monitoring project progress.
- (Software Architect) Designing the overall software architecture and ensuring it aligns with project requirements.
- (Software Architect) Collaborating with the development team to guide the implementation of the software architecture.
- (Software Architect) Identifying and addressing software design issues.

Özde (Business Analyst and Tester):
Responsibilities:
- Elicitation and documentation of business requirements from stakeholders.
- Analyzing and documenting current business processes and proposing improvements.
- Collaboration with stakeholders to ensure requirements align with business goals.
- Bridging communication gaps between technical and non-technical team members.
- Preparation of use cases, user stories, and other documentation for the development team.
- Ensuring the software is thoroughly tested and meets quality standards before release.
- Defining and implementing testing strategies, including unit testing and integration testing.

Elif (Software Developer and Software Architect):
Responsibilities:
- Writing high-quality, efficient, and maintainable code.
- Collaborating with the software architect to understand the system architecture and design.
- Participating in code reviews and providing constructive feedback.
- Debugging and resolving software defects.
- Adhering to coding standards and contributing to the development of best practices.
- (Software Architect) Designing the overall software architecture and ensuring it aligns with project requirements.
- (Software Architect) Collaborating with the development team to guide the implementation of the software architecture.
- (Software Architect) Identifying and addressing software design issues.

2.2. Communication Channels
- Whatsapp for communication
- Google Meet for meetings
- GitHub for sharing the documentation and codes

3. Project Practices and Measurements

The project will use an iterative agile development approach which permits incremental updates and improvements. In order to track progress, we will have iteration assessments to provide regular reviews and retrospectives at the end of each iteration. Additionally, we will track progress by collecting the velocity per iteration (completed work item points/iteration). Continuous integration will also be implemented to ensure that changes to code are integrated and tested regularly. The metrics that will track continuous integration include code coverage, build success/failure rates, and test pass rates. Independent Testing will be employed to ensure software quality and will be measured using defect reports and test case pass rates. Regarding source code version control, GitHub will be utilized to manage source code changes and tracked using commit history and branching statistics.

4. Project Milestones and Objectives

4.1. Project Milestones and Objectives for Release 1

Iteration	Primary Objectives (risks and use case scenarios)	Scheduled Start or Milestone	Target Velocity (Story Points)
I1	**Objective:** Project Initiation and Planning (Inception Phase) 1. Project Initiations 2. Define Project Plan, Risk List and Work Item List Documents 3. Define Use Cases and their Scenarios 4. System-wide Requirements Specification Documents **Risks:** 1. Mitigate Risk 1	23.10.2023 - 4.11.2023	38
I2	**Objective:** Technical Research and Implementation of Use Case 5 - 9 (Elaboration Phase) 1. System Design and Architecture, UI and Database for Use Case 5 - 9 2. Implement Use-Case 5: Manage User Account 3. Test Use-Case 5: Manage User Account 4. Implement 9: Manage Service 5. Test Use Case 9: Manage Service **Risks:** 1. Mitigate Risk 1 2. Mitigate Risk 6 3. Mitigate Risk 8 4. Mitigate Risk 7	6.11.2023 - 25.11.2023	53
I3	**Objective:** Implementation of Use Case 2 (Elaboration Phase) 1. UI and Database for Use Case 2 2. Implement Use-Case 2: Search Service 3. Test Use-Case 2: Search Service **Risks:** 1. Mitigate Risk 1 2. Mitigate Risk 6	28.11.2023-15.12.2023	32
I4	**Objective:** Implementation of Use Case 1-12 (Construction Phase) 1. UI and Database for Use Case 1, 12 2. Implement Use-Case 1: Schedule Appointment 3. Test Use-Case 1: Schedule Appointment 4. Implement Use-Case 12: Approve Appointment 5. Test Use-Case 12: Approve Appointment 6. User acceptance testing **Risks:** 1. Mitigate Risk 1 2. Mitigate Risk 6	16.12.2023-02.01.2024	34
I5	**Objective:** Deployment (Transition Phase) 1. Deploy software on production environment 2. Configure software for hardware **Risks:** 1. Mitigate Risk 1 2. Mitigate Risk 7	03.01.2024-20.01.2024	8

	3. Mitigate Risk 8		

4.2. Project Milestones and Objectives for Release 2

Iteration	Primary Objectives (risks and use case scenarios)	Scheduled Start or Milestone	Target Velocity (Story Points)
I1	**Objective:** Implementation of Use Case 4-6 (Construction Phase) 1. UI and Database for Use Case 4-6 2. Implement Use-Case 4: Manage Proposal 3. Test Use-Case 4: Manage Proposal 4. Implement Use Case 6: Ask Help 5. Test Use Case 6: Ask Help **Risks:** 1. Mitigate Risk 1	21.01.2024-15.02.2024	32
I2	**Objective:** Implementation of Use Case 8-10 (Construction Phase) 1. System Design and Architecture, UI and Database for Use Case 8-10 2. Implement Use-Case 8: Make Proposal 3. Test Use-Case 8: Make Proposal 4. Implement Use-Case 10: Receive Payment 5. Test Use-Case 10: Receive Payment **Risks:** 1. Mitigate Risk 1 2. Mitigate Risk 6 3. Mitigate Risk 10	16.02.2024-29.02.2024	34
I3	**Objective:** Implementation of Use Case 3-11 (Construction Phase) 1. UI and Database for Use Case 3-11 2. Implement Use-Case 3: Make Payment 3. Test Use-Case 3: Make Payment 4. Implement Use-Case 11: Troubleshoot System 5. Test Use-Case 11: Troubleshoot System **Risks:** 1. Mitigate Risk 1 2. Mitigate Risk 6 3. Mitigate Risk 10	1.03.2024-17.03.2024	20
I4	**Objective:** Implementation of Use Case 7 (Construction Phase) 1. UI and Database for Use Case 7 2. Implement Use-Case 7: Write Review 3. Test Use-Case 7: Write Review 4. User acceptance testing	18.03.2024-30.03.2024	32

	Risks: 1. Mitigate Risk 1 2. Mitigate Risk 6		

5. Deployment

Our deployment strategy for this project adopts an agile and iterative methodology, allowing for flexibility and responsiveness to evolving requirements. The development stack comprises JavaScript and Node.js, providing a robust foundation for building dynamic and scalable applications. In the initial release, we will create a standalone application utilizing MySQL for database object definition and creation, ensuring data integrity and efficiency. This standalone version will undergo continuous development and testing, aligned with agile principles, to swiftly deliver a functional product. For continuous integration and development, GitHub will serve as the primary platform for updating and managing the source codes.

In the subsequent release, our focus will shift towards cloud deployment, and we have chosen Google Cloud as our primary cloud service provider. Leveraging Google Cloud services, we will enhance the application's availability and scalability while minimizing initial investment. The microservices architecture will be implemented using Docker containers for containerization, Kubernetes for orchestration, and Istio for service mesh functionality. These tools, tailored for Google Cloud, will enable independent development and deployment of application components, ensuring seamless scaling and efficient management. This approach helps to translate a standalone application to a Google Cloud-hosted solution, embodying our commitment to deliver a dynamic and adaptable software product, meeting the evolving needs of our users.

6. Lessons Learned

At the end of each iteration, retrospectives and regular reviews will be performed to evaluate project progress, address issues, and identify improvement strategies. We will track progress by collecting the velocity per iteration (completed work item points/iteration). The lessons learned from the retrospective will be recorded in this section as the iterations progress. The topics for lessons learned may include evaluating the build and deployment times, availability of reliable test data, communication among the team, sprint planning efficiency, availability of documentation, and clarity of the scope for each iteration.

6.1 Iteration 1

- Communication: Using polls on WhatsApp was an effective and efficient way to coordinate meeting availability and times.
- Work Distribution: Assigning similar tasks to the same individuals was helpful for maintainability and internal consistency.
- Work Products: We need to sit together to review every document for their internal and external consistency.

6.2 Iteration 2

- Resources: Because we lost a teammate at the end of Iteration 2 (a decrease in resources), we needed to change the scope of the remaining Iterations.
- Work Distribution: There was a significant learning curve for the remainder of the team to re-distribute the assignments of the lost teammate due to areas of specialization. We need to consider this for resource allocation and Iteration planning.
- Work Products: Although all the documents and work products work together for the cohesive and iterative development of the project, the Project Plan and Work Item List were key documents for determining the scope of each Iteration and we frequented these documents often.

6.3 Iteration 3

- Architecture: It may be necessary to re-evaluate the original architectural decisions to ensure they can continue to support the development of the use cases.
- Implementation: The Domain Models, Use Cases, and Interaction/Sequence Diagrams are vital to understand and design the system properly and consistently.