SOFTWARE PROJECT MANAGEMENT PLAN

High Performance Cache

Version 1.0

Change History

Revision	Change Date	Description of changes
V1.0	02/05/18	Initial release

Table of Content

I. Introduction

The present document is the Software Project Plan for the "High Performance Caching of Travel Inventory". This project is being developed by team 4. The content of this document is divided into four parts:

i. Project Purpose and Scope
i. Project Structure and Organization
ii. Project management and Control
iii. Technical Process
iv. Project Schedule and budget

Team 4

Name	Email	Phone
Robert Montgomery	rmontg16@students.kennesaw.edu	(708) 601-0095
Jarred Wilson	jwill836@students.kennesaw.edu	(404) 805-9427
Malcolm Frank	malcolmfrank91@gmail.com	(404) 798-3864
Macauley Odinaka	modinaka@students.kennesaw.edu	(404) 662-8654
Dickson Diku	ddiku@students.kennesaw.edu	(678) 876-6599
Abebe Adamu	abbkassa@gmail.com	(404)563-9988

II. Project Objective and Scope

1. Overview

Travelport is a travel e-commerce platform that enables travel providers, agencies, corporations, and developers to search, share, buy, and sell travel. Travelport carries out this service through their universal API. The Travelport Universal API offers an array of content for air, hotel, car, and rail, including ancillaries. It also provides functionality to build complete traveler, agency, branch, and account profiles. Currently, travelport is undertaking an effort to improve their methods of caching inventory data used during search processes by removing single points of failure in the existing application.

The goal of this project is to highlight and eliminate potentially interruptive processes within the current caching system that could inhibit performance. We intend to accomplish this goal by developing behavioral and non-behavioral requirements for a cache application that can scale to 100 billion requests per month at associated peak usage, including adding data to the cache, accessing data from the cache, and developing a solution architecture that eliminates the single points of failure associated with a number of in-memory database technologies. In pursuit of that goal, the team will work in coordination with the sponsor organization to continually verify the project's progress and direction. To that end, the team will prepare and deliver several documents for sponsor approval that describe the functionality and design of the software. The sponsor will also receive all code and any other pertinent deliverables included in the finished project.

2. Purpose

We will design and develop a proof of concept application using Java Caching System (memory cache LRU) and in-memory database technologies (Hazelcast) to demonstrate its ability to scale and meet response time and fault tolerance requirements with appropriate load and failure test scripts. Our proof of concept application will provide an adequate solution architecture in which Travelport can implement to modify/improve their current caching system.

3. Scope

he scope of this project is to fasten the customer search experience. This means that the ecommerce services response time will be improved by storing data in memory to reduce time required to access it from disk.

This document contains a plan for the successful execution of the goals that will be outlined further defined in the Software Requirements Specification (SRS) document.

4. High Level Functionalities

a. Retrieve inventory data from a database

b. Add retrieved data to the cache

c. Access data from the cache

d. Scale to support one hundred billion requests per month

e. Store cache in RAM

f. Meet response time requirement

g. Handle failures to meet fault tolerance requirements

5. Deliverables

The following table outlines the various documents associated with this project:

Document	Description	Due Date	Breakdown
Software Project Plan	Extended overview over the full project and the steps intended to complete the project.	02/05/2018	Each section of the project plan will be divided amongst the entire team, so that we can ensure that two or more team members are not working on the same section.
Software Requirement Specifications	Lays out functional and non-functional requirements and use cases that describe user interactions that the software must provide.	02/19/2018	Each section of the SRS will be divided amongst the entire team, gathering function and non-functional requirements based on the client requirement
Software Design Specification	Describes all data, architectural, interface and component-level design for the software.	03/05/2018	Each section of the software design specification will be divided amongst the entire team to ensure that all interface and component-level design is covered.
Executable code/Source code	Any and all code that is needed to implement in the project.	03/19/2018	All team members will be involved in the software development portion, this includes coding and prototyping any code needed to implement our complete design.
Test Plan	A plan developed to test the software.	03/26/2018	Each section of the test plan will be divided up evenly amongst the team members, to develop how the

			software should be tested.
Test cases Specification	Specification that displays the test cases of the software.	04/02/2018	Half of our team members will create different test cases of the software and display their information in the Test Cases Specification document.
User Guide	Displays the instructions on how to properly use the software.	04/02/2018	Another half of our team members will create a user guide on how the software is supposed to properly function.

6. References

IEEE Std 1058-1998, IEEE Standard for Software

Project Management Plans.

7. Definitions and Acronyms

SMPP: Software Management Project Plan- The purpose of this document is to provide a detailed plan and timeline in which we mean to implement a high performing caching systems.

SRS: Software Requirements Specification- The purpose of this document is to provide functional and nonfunctional requirements for our intended software system. It also contains use cases that describe user interactions that the software must provide.

SDS: Software Design Specification - The purpose of this document is to provide a detailed architecture for which we intend to implement our high performing caching system. This document will also include use cases and other

STP: Software Test Plan - The purpose of this document is to describe the testing scope, approach, resources, and schedule of intended test activities. It identifies amongst other test items, the features to be tested, the testing tasks, who will do each task, degree of tester independence, the test environment, the test design techniques and entry and exit criteria to be used, and the rationale for our choice, and any risks requiring contingency planning.

API: Application Programming Interface - a set of subroutine definitions, protocols, and tools for building application software; a software intermediary that allows two applications to talk to each other.

IMDB: In-Memory Database - is a database management system that primarily relies on main memory, rather than disk space, for computer data storage.

Cache: Cache memory is a small-sized type of volatile computer memory that provides high-speed data access to a processor and stores frequently used computer programs, application, and data. It is the fastest memory in a computer, and is typically integrated onto the motherboard and directly embedded in the processor (disk caching) or main random access memory (memory caching).

RAM: Random Access Memory - is a form of computer data storage that stores data and machine code currently being used. A RAM device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory.

Disk Storage: is a general category of storage mechanisms where data is recorded by various electronic, magnetic, optical, or mechanical changes to a surface layer of one or more rotating disks. A disk drive is a device implementing such a storage mechanism.

SDLC: Software Development Life Cycle - a well-defined, structured sequence of stages in software engineering to develop the intended software product. SDLC provides a series of steps to be followed to design and develop a software product efficiently. SDLC framework includes the following steps:

- Communication - user initiates requests for desired software product
- Requirements Gathering - The team elicits information from the stakeholders to better understand their requirements for the software system. The requirements are contemplated and segregated into user requirements, system requirements, and functional requirements.
- Feasibility Study - when the software team analyzes if a software can be made to fulfill all requirements of the user and if there is any possibility of the software becoming useless.
- System Analysis - A this step developers decide a roadmap of their plan and try to bring up the best software model suitable for the project. System analysis includes understanding the software product limitations, learning system related problems or changes to be done in existing systems beforehand, identifying and addressing the impact of project on organization, personnel, and etc.
- Software Design - Combining the information gathered from the requirements and analysis phases to create a viable solution architecture for a desired software system. The inputs from users and information gathered in the requirements phase are the inputs of this step. The output of this step comes in the form of two designs: logical design and physical design. Engineers produce metadata and data dictionaries, logical diagrams, data-flow diagrams and in some cases pseudo-code.
- Coding - The implementation of software design starts in terms of writing program code in the suitable programming language and developing error-free executable programs efficiently.
- Testing - Software testing is done while coding by the developers and thorough testing is conducted by testing experts at various levels of code such as module testing, program testing, product testing, in-house testing and testing the product at user's end. Early discovery of errors and their remedy is the key to reliable software.

- Integration - Software may need to be integrated with the libraries, databases and others program(s). This stage is involved in the integration of software with outer world entities.
- Implementation - This means installing the software on user machines. At times, software needs post-installation configurations at user end. Software is tested for portability and adaptability and integration related issues are solved during implementation.
- Operation and Maintenance - This phase confirms the software operation in terms of more efficiency and less errors. If required, the users are trained on, or aided with the documentation on how to operate the software and how to keep the software operational. The software is maintained timely by updating the code according to the changes taking place in user end environment or technology. This phase may face challenges from hidden bugs and real-world unidentified problems.
- **GDS**: Global Distribution System - a computerized network system owned or operated by a company that enables transactions between travel industry service providers, mainly airlines, hotels, car rental companies, and travel agencies.
- **JCS**: Java Caching System - a distributed caching system written in Java. It is intended to speed up applications by providing means to manage cached data of various dynamic natures.
- **Eviction Policy**: method to predict which entries within the cache are most likely to be used again.
- **LRU**: Least Recently Used - an algorithm that assumes that based on the observation that events (data within the cache) have been heavily used in the last few instructions will probably be heavily used again in the next.

III. Project Structure and Organization

1. Process Model

We will use the Waterfall Model for two-week period and with 6 iterations, known as Phase 1.x ,2.x, 3.x,4.x,.x, and 6.x. In each iteration we will develop a requirements specification document as well as a prototype to demonstrate possible solutions.

2. Structure

Name	Role	Responsibilities
Dickson Diku	**Team Leader**	
Abebe Adamu	**Business Analyst**	**Requirements elicitation. Collaborate with Solution Design Architect to create SDS.**
Jarred Wilson	**Lead Solution Design Architect**	**Create solution design architecture to be implemented for the desired system. Communicate design criteria with Software Developers to ensure that all design goals understood and met during implementation.**
Robert Montgomery	**Lead Software Developer**	**To implement the application and make sure that all requirements are met**
Macauley Odinaka	**Lead Tester Engineer**	**Ensure that the functionalities of the application works efficiently.**
Malcolm Frank	**Technical Writer**	**Ensure that all documentation is correct and consistent throughout SDLC. Follow-up with other team members to ensure that project/system requirements and guidelines are properly communicated and prevalent throughout SDLC.**

3. Project Roles and Responsibilities

Project phase	Description	Phase Lead
System Concept Development Phase	Planning the system concept, gathering all tools needed to produce the software.	Dickson Diku
Planning Phase	Create a plan in order to develop a solution for our client.	Dickson Diku
System Requirements Analysis	Analyze any and all requirements primarily inside the SRS.	Everyone
Design Phase	Designing how the software should look.	Jarred Wilson, Dickson Diku, Robert Montgomery
Implementation Phase	Using our developed software to implement a solution to the client's problem.	Robert Montgomery, Jarred Wilson, Dickson Diku
Quality assurance and testing	Testing software for quality assurance.	Macauley Odinaka, Malcolm Frank, Abebe Adamu
Documentation	Ensuring all deliverables meet standards.	Malcolm Frank, Dickson Diku, Jarred Wilson

IV. Project Management and Control

1. Management Objectives and Priorities

The management objective is to deliver the product on time and of high quality. The Team Leader and all team members work together to achieve this by respectively checking that progress is made as planned and monitoring the quality of the product at various stages.

2. Assumptions, Dependencies and Constraints

Assumptions: Caching system will be platform independent. The performance goals are achievable with currently available hardware. The current infrastructure cannot be modified to achieve the project goals.

Dependencies: The use of existing in-memory database technologies to retrieve data to be cached.

Constraints: Cache must be stored in RAM.

3. Risk Management

Risks for this project have been classified accordingly.Major risks we can determine for this software are as follows:

It would appear that “insufficient research on technical requirement information” is a significant issue, given the estimates according to our estimation

Risks	Risk Level	Probability	Impact	Risk Mitigation Strategies
Insufficient research on technical requirement information	high	60%	1	The team will gather and prepare more research and relevant questions before organizing additional meeting with the clients to make sure that more technical informations were researched and understood.
Inappropriate version of tools and components	High	40%	1	Select and compare specific versions of tools and components to use and every member will adhere to the choice throughout the entire project.
Failure to meet deadline for deliverables	High	30%	1	Setup milestones in advance of the final due date for each deliverable.

Changes in requirements	High	30%	2	Client will be made aware, in advance, of the amount of change that can be accommodated within the term of the project.
Insufficient Testing	Minor	10%	3	Design an effective testing strategy. Allow for sufficient time for unit testing, system testing , performance testing, integration testing and user testing.
Accidental loss of valuable information	Minor	50%	3	The project will be done using GitHub, including version control software, and the resulting documents stored online to minimize the chances of loss of information. In addition, each team will keep copies of work that are not stored online.
Poor code's comments	Minor	30%	4	Programmers need to follow a good documentation style for each code they are writing. Team Leader will check for clarity and cleanness of the comments.

1: Critical

2: Significant

3: Normal

4: Less

4. Project Monitoring and Controlling

FreedCamp is used to track the progress of the project and the periodic status meetings arranged in the team. Every week developers report to the manager about the technical details. We will be using the Scrum methodology in order to complete the project in a timely manner.

V. Technical Process

The technical process model describes the methods that the team will use in representing the technical details of the project. In addition, the technical process outlines how the team will record and publish technical details during project development, as well as the version control process that will be used.

1. Methods, Tools, and Techniques

The High Performance Caching of travel inventory is developed using Scrum Master process methodology.Our developers are required to use the Eclipse IDE 4.7 to minimize issues with cross compatibility while developing. This allows the use of Eclipse extensions (if supported by TravelPort API). Redis will be used for in memory database technology. UML diagrams will be generated using the Eclipse plugin: UML2 Extender SDK. This will also be used to define the process. Visual Paradigm 14.1 will also be used as UML modeling tool that supports iterative development, which will support the transition for each phase. Microsoft Visio may also be used for simpler UML diagrams.

2 Software documentation
During the project, documents should conform to a number of aspects:
Documents must be of good quality. The standards all documents are required to meet are documented in the SCMP [7] with respect to style and in SQAP [8] with respect to content.
Documents must be reviewed. The manner in which document reviews are performed is described in
the SVVP [13].
The purpose of document reviews is to get docs of high quality.

3 Project support functions
Besides Project Management, three other management functions are present:

Configuration Management:

The purpose of software configuration management is to plan, organize, control and coordinate the identification,storage and change of software through development and transfer.

Verification and Validation.

Software Verification and Validation activities check the software against its
specifications. The Quality Manager writes the SVVP as one of his tasks. In it he outlines how he plans to
perform his verification and validation activities.

Quality Assurance.

During the project, all documents should be tested against quality standards; both to conform to the chosen project documentation style and to contain correct information of reasonable
quality.

VI. Project Schedule and Budget

1. Activities and Tasks

Activities and tasks will be updated periodically as the development team progresses through the software development process inside our weekly reports.

2. Schedule

Deliverable	Due Date
SPMP approved	Monday, February 5, 2018
SRS approved	Monday, February 19, 2018
SDS approved	Monday, March 5, 2018
Test Plan approved	Wednesday, March 26, 2018
STS approved	Wednesday, April 2, 2018
UG and Executable code	Wednesday, March 19, 2018

3. Budget and resources Allocations

the project will be implemented over 12 weeks or 2016 hours

Hardware/Software	Unit Cost $	Quantity	Total Cost $
IntelliJ IDE	free	6 licences	0.00
Microsoft Office Suite	free	6 licences	0.00
Laptops		6	
FreeCamp		1 licence	
Swagger.io	free	1 licence	
Postman	free	6 licences	

Labor	Budget Estimated (man-Hours)
SMPP	100.00
SRS	200.00
SDS	200.00
Implementation	916.00
STS	300.00
System Test Report	100.00
User Guide	100.00
Software Maintenance Manual	100.00
Total	2016.00

VII. Approvals

Name	signature	Date
Dickson Diku	DD	02/05/2018
Robert Montgomery	RM	02/07/2018
Malcolm Frank		
Jarred Wilson		
Abebe Adamu Kassa		
Macauley Odinaka		
Leonard Gresky		