4.0: *Software Development Plan*

4.1 Plan Introduction

This Software Development Plan provides the details of the planned development process for the distributed database application which stores and retrieves data using decentralized, peer-to-peer principles. The application can be broken into three main subsystems: the peer-to-peer network, blockchain, and proof of stake consensus protocol. The peer-to-peer network will allow nodes to securely pass messages to other nodes in the system. The blockchain allows data to be securely stored, accessed, and managed, while maintaining a complete system history. The proof of stake consensus protocol allows the blockchain to be distributed amongst nodes and the state to be managed in the decentralized system. In order to complete the application at hand, the following tasks listed in the project deliverables must be completed with success.

4.1.1 Project Deliverables

- Week 2
  - Project Proposal Document
    - A high-level overview of the application that is going to be developed, the primary features, the incentive behind the project, expectations and more of the basic behind the project
- Week 4
  - Proof of Stake/Work Research
    - Have complete research on a wide-breadth of PoS implementations
  - Design Proof of Stake Validation Approach
    - Create preliminary plan for PoS system with support for future development
  - Learn Golang
    - Gain a proficient level of knowledge of the programming language Go
    - Have a strong understanding of the native package "libp2p" to gain access peer-to-peer functions and objects
- Week 5
  - Software Development Plan Document (Initial)
    - A detailed document describing the process of creating the application
    - Will establish deadlines and expectations of when certain aspects of the application and documentation are to be completed
  - Requirements Specification Document
    - This document establishes the tools needed to create the application at hand and allows the client to be aware of all of the necessities
  - Design Golang structs for nodes

- Layout all required structs to allow implementation team to work more efficiently

- Week 6
  - Program Golang structs
    - Create the designed structs allowing the team to work without creating redundant structs
- Week 7
  - Software Development Plan Document (complete)
    - Any needed changes to the Software Development Plan Document that were not implemented in the initial submission
- Week 8
  - Program Blockchain
    - Complete the modification on the of-the-shelf blockchain implementation, offering support for our transactions
- Week 9
  - Program Peer-To-Peer model
    - Complete peer-to-peer communication system, allowing data to be securely sent between nodes
- Week 11
  - Software Design Description Document (Architecture)
    - The overall architecture of the software system as well as the internal interfaces
- Week 12
  - Project Critical Design Review Presentation
    - Scrum session to detail and lockdown the full details of the SDD
  - Software Design Description Document (complete)
    - Update the details of the application's design to the initial architecture submission
- Week 13
  - Program Proof of Stake Validation
    - Implement proof of stake consensus protocol with designed protection from malicious attacks
  - Program Encryption
    - Complete blockchain encryption and signature system
- Week 14
  - ALPHA/BETA Presentation/Demonstration
    - Have a working version of the system running on multiple nodes, with basic functionality
  - Test and Integration Plan
    - Tests laid out along with applicable methods of running the test suite
- Week 15
  - User's Manual Final Updates
    - Complete the documentation along with quick start guide for setup and usage
- Week 16
  - FINAL Project Presentation

        - Create a poster displaying an overview of the application, technologies that are utilized, and more to allow guests to understand the project
- Ongoing
    - FINAL Product Deliver (Final Report and Code)
        - Completed MVP with support for deployment as a distributed, decentralized system with support for adding, retrieving, and modifying data

4.2 Project Resources

Listed down below are the hardware and software resources required for the success of the project.

4.2.1 Hardware Resources

Development Machine

Category	**Requirement**
Processor	Any Intel Pentium Processor
RAM	128 MB
Hard Drive Space	256 MB
Display	Any

Client Machine

Category	**Requirement**
Processor	All 64-bit x86 processors , any Pentium MMX or later processor
RAM	64 MB
Hard Drive Space	128 MB
Display	Any

4.2.2 Software Resources

Development Machine

Category	Requirement
CLI	Bash
Operating System	Mac OS X Yosemite 10.10 or newer
Text Editor	Any

Client Machine

Category	Requirement
CLI	Bash
Operating System	Mac OS X Yosemite 10.10 or newer
Text Editor	Vim

4.3 Project Organization

Debased splits its members into three roles. J Goocher is primarily responsible for the design and conceptual aspect of the project. Having a strong background in the fundamentals of modern blockchain algorithms, J leads debased by brainstorming the optimal algorithms for the project and presents his ideas to the group. Following that, he runs brainstorming sessions to ensure the best approach is being developed from a design perspective. Thomas O'Brien focuses on the Golang implementation aspect of debased. Previously working with Golang during his summer internship and for his CMSI 402 senior project, he has developed a knack for Golang. Thomas supports Debased in helping the team understand the fundamentals of Golang. Not only that, but Thomas leads on the code base design for debased itself. He conceptualizes over the best approach for programming an efficient decentralized debased while still maintaining the integrity of the industry standard set for Golang code. Jackson Watkin's role revolves over his excellent skills as a programmer. His main focus is leading the teams in the writing of the Golang code itself and supporting his team on debugging various issues. Not only that, but he is central to the overall organization of the team's meetings and acts as the SCRUM master. While each team member has their respective role, everyone on the team supports one another with blockers, takes part in other teammate's focus, and overall wears multiple hats.

4.4 Project Schedule

This section provides schedule information for the debased project.

4.4.1 GANTT Chart

Refer to the file **gant-chart.xlsx** or **gant-chart.pdf** within **documents/** of the debased repository.

4.4.2 Task/Resource Table

Refer to the file **task-resource-table.xlsx** or **task-resource-table.pdf** within **documents/** of the rebased repository.