****

PROJECT PLAN

***Wealth Weaver – Finance Management Tool***

|  |  |  |
| --- | --- | --- |
| **Date** | 13-02-2025 | |
| **Version** | 0.1 | |
| **Status** | *Draft* | |
| **Author(s)** | Balan Mihai-Stefan | 4748565 |

A blue logo with a shark head

Description automatically generated

#### Version control

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Author(s) | Amendments | Status |
| 0.1 | 13-02-2025 | Balan Mihai | Template created | *Complete* |
| 0.2 | 24-02-2025 | Balan Mihai | Completed plan | *Complete* |

#### Communication to client

|  |  |  |
| --- | --- | --- |
| Version | Date | To |
| 00.1 | ` | Stakeholders |

Table of contents

1. Project Assignment 4

1.1 Project Context 4

1.2 Project Goal 4

1.3 The assignment 4

1.4 Scope 4

1.5 Conditions 5

2. Approach and Planning 7

2.1 Approach 7

2.2 Research methods 7

2.3 Breakdown of the project 9

3. Project Organization 10

3.1 Team members 10

3.2 Communication 11

4. Finance and Risks 12

4.1 Cost budget 12

4.2 Risks and fall-back activities 12

# Project Assignment

## Project Context

In today’s world finance is becoming an increasingly more complicated topic. Studies show that more and more people are unaware of their spending habits and are unable to reach their financial goals, understand how their taxes work and many other crucial aspects of modern life. Wealth Weaver is an application made to help the average person understand and manage their personal finance, motivate friends, set goals and stick to them in the long term.

## Project Goal

**The result of the project is a user-friendly application designed for the average person, which is of the highest quality possible. The app is always available, without experiencing performance drops, regardless of the number of users active. Using the platform from anywhere on the globe is not an issue, as all currencies are supported.**

## The assignment

**The main objectives of the project are:**

* **To design and implement a robust back end distributed software system that can handle any situation**
* **To create a minimal user interface that provides the functionality clearly to the user.**
* **To integrate with external APIs that can enhance the user experience and provide useful information and services.**
* **To offer a dependable, scalable and secure data storage system that can manage different types of data.**
* **To implement innovation leading cloud and deployment technologies such as Kubernetes, Docker, auto-scaling services and load balancing services.**

## Scope

|  |  |
| --- | --- |
| **The project includes:** | **The project does NOT include:** |
| Secure user management service and secrets storage | 24/7 available service regardless of the number of active users being 0. |
| Safe read-only linking of bank accounts to provide a history of transactions automatically to the app. | Fully automated cloud deployment through tools such as Terraform |
| Insight provisioning for spendings |  |
| Demand-based automated scaling of individual services. |  |
| Distributed data storage and efficiency when dealing with a large amount of data |  |

## Conditions

|  |  |
| --- | --- |
| **Condition** | **Explanation** |
| **Scalability** | * Use of Kubernetes and Horizontal Pod Autoscaler for each service. Pairing such a system with a client-facing Layer 4 load balancer, for streamlined user monitoring and added security. |
| **Cost-effectiveness** | * Usage of Azure Function Apps and FaaS deployment practices to reduce wasting resources when not a small group of users is using the platform. Integrating cost-per-use instead of cost-per-uptime. |
| **Reliability and availability** | * Leveraging Cloud Services to maintain of services while they are in demand, scale individually, based on user demand. |
| **Disaster recovery and backup** | * Use of distributed data to avoid SPO datacentres. Have the option to always load backup. |
| **Security** | * Implementing strong network segregation, limiting user actions, connections, and avoiding discovery of internal services and structure. Using Layer 4 load balancing and having singular NAT bridge per user. |
| **Futureproofing** | * Designing software systems in an expandable way and leveraging architectural principles to make upgrading regression less and versioning seamless. |

# Approach and Planning

## Approach

The approach chosen in planning and developing the project will be the SCRUM / Agile method. […]. The length of an agile sprint will be **3 weeks** with every sprint ending in a demonstration of the work done inside the sprint to the project coach, collection of feedback and retrospective on what succeeded during the sprint and what did not. The last day of a sprint should be reserved to compile a list of additional requirements/suggestions from the coach that will help the fine-tuning of the following sprints.

### Problem definition

The problem definition phase, also known as “sprint 1” is aimed at understanding the context of the project and coming up with a well-structured plan for a solution. During “sprint 1” preliminary feedback sessions with the technical teachers have been set-up, which will allow the team to plan the future sprints, after determining the urgency of each matter in question.

### Test approach

Testing will be conducted after the implementation of every major feature in the project, as well as one final testing before the end of every sprint to make sure that the product (MVP) which will be demonstrated to the client is working as planned.

### Completion phase

The purpose of the completion phase, or the “last sprint”, is to allow enough time for the implementation of extra features, debugging and fine tuning of the project, as well as preparing a comprehensive presentation to accompany the deployment of the final product, as specified in the project guidelines.

## Research methods

### Problem definition phase

### Understanding the context of the problem:

* + **Document analysis** (**FIELD**) – studying the project reader and defining clear context, audience of project, requirements and constraints.
  + Stakeholder analysis (**FIELD**) – establishing the stakeholders and taking their requests into account when designing our solution.

### Creating a well-structured plan:

* + **Brainstorming** (**WORKSHOP**) – thinking of solutions for the problems raised, as well as their implementations.
  + **Business case exploration** (**WORKSHOP**) – analysing most viable scenarios for every problem and opportunity.
  + **Available product analysis** (LIBRARY) **–** studying previous project plans, as well as open-source information about similar cases/problems and their solutions.
  + **IT architecture sketching** (**WORKSHOP**) – drawing a network diagram that is descriptive enough to provide support during the development phase.
  + SWOT analysis (LIBRARY) – analysing the features that are planned for implementation to determine their feasibility and impact on the final product.

### Agile sprints

### Planning phase:

* + **Requirements prioritization** (**WORKSHOP**) – planning sprints so that the most important requirements are met first, in order to minimize risk associated with lack of time.
  + **Multi-criteria decision making** (**WORKSHOP**) – taking all information available into account, then dividing the work volume among team members.
* Development phase:
  + Computer simulation (**LAB**) – virtualizing and testing the solution before implementation (through software such as GNS3 and locally-hosted VMs).
  + Unit test (LAB) – testing a piece of code before pushing it to the main repository.
  + Component test (**LAB**) – testing a network device/branch of the infrastructure individually.
  + Non-functional test (**LAB**) – checking whether the services have enough capacity for the predicted workload or not.
  + System test (LAB) – testing the system after implementing an important feature.
* Presenting and reflection phase:
  + Peer review (SHOWROOM) – obtaining feedback from teammates, classmates and teachers.
  + Product review (SHOWROOM) – testing the product before presenting our work to the client.
  + Pitch (SHOWROOM) – presenting the product to the client throughout all its development stages.

## Breakdown of the project

|  |  |
| --- | --- |
| Sprint | Sprint goals |
| Sprint 0 |  Establish ideation and integrate suggestion in planning  à Develop a draft of the Project Plan.  à Develop a research plan  à Make architectural choices and document them in a separate explanatory document. |
| Sprint 1 |  Architecture Design and preparation  à Implementation of software application at the wireframe level  à Analyse and integrate feedback into the SDL of the project |
| Sprint 2 | à Address any spillover from previous sprint  à Improve on technical implementation, software-wise.  à Automated component testing and compatibility testing with selected third-party providers.  à Complete research on banking systems |
| Sprint 3 | à Address any spillover from previous sprint  à Finalize software-wise implementation and conduct a successful demo  à Service decoupling full-scale test and performance test.  à Finalize research on deployment and DevOps side of project and create a detailed plan for next sprint |
| Sprint 4 | à Address any spillover from previous sprint  à Begin local deployment according to conducted research  à Get an MVP of the local (testing/acceptance) environment  à Address security and GDPR concerns in testing environment, prepare for live deployment |
| Sprint 5 | à Address any spillover from previous sprint  à Begin deployment to production - Azure Cloud - of the application on free trial subscription  à Stress testing for load balancing and auto scaling in testing/acceptance environment  à Address findings of stress testing |

# Finance and Risks

## Cost budget

Free trials for Azure Cloud for Students should be employed. This allows for 100 euros credit and one month of uptime. Due to this constraint, deployment is scheduled for end of semester (see 2.3 Breakdown of the project)

Due to the limited amount of allocated credits deployment machines will be chosen to optimize for cost effectiveness rather than best performance available. The decision will be further explained in the Architecture Document. In addition to that, no form of load or stress testing will be conducted in the production environment, due to the chance that credits may be exhausted.

## Risks and fallback activities

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Risk** | **Prevention activities included in plan** | **Fall-back Activities** |
| **1.** | Project delays duo to unforeseen circumstances. | Allocate additional time in the project timeline to account for potential delays and have a contingency plan in place for critical path activities. | Communicate delays and reasons with stakeholders and adjust planning accordingly. |
| **2.** | Scope creep | Clearly define the project scope and ensure that all stakeholders are aligned on it. Monitor the project closely to ensure that it stays within scope and be prepared to adjust as needed. | Rediscuss priority, adjust features to be developed according to the must-have features agreed upon with the stakeholder. |
| **3.** | Failure of external service provider | Create the system with decoupling in mind. This way, if one external service provider has an outage, other parts of the system can still function. | Establish communication with a representative of the external provider’s development or support team to mitigate the issue. |
| **4.** | Lack of communication or collaboration among project team members | Establish clear communication protocols and expectations for the project team. Conduct regular check-ins and progress updates to ensure that everyone is aligned on the project goals. | We are going to implement a communication and collaboration plan that promotes effective communication and teamwork. |
| **5.** | Unforeseen hardware or software failures. | Implement redundancy and backup systems to minimize the impact of hardware or software failures. | Revert to the latest back-up when software failure occurs. |
| **6.** | Network Connectivity Issues | Test network connectivity and bandwidth requirements before the implementation. | Use alternative network connections or postpone the migration until network issues are resolved. |
| **7.** | Downtime | Plan the implementation during off-peak hours to minimize impact. | Communicate downtime to users in advance, provide maintenance pages or temporary solutions. |
| **8.** | Security Breach | Implement strong access controls, encryption, and security measures for data in transit and at rest. | Isolate compromised systems, investigate the breach, and restore from clean backups. |