

Project Report for <Project Name>

Practice Module for Certificate in Architecting Scalable Systems

Team <Number>

Members:

<Member’s Names>

CONTENTS

1. Introduction 4

1.1 Background 4

1.2 Business Needs 4

1.3 Stakeholders 4

1.4 Project Scope 4

1.4.1 Functionality in scope 4

1.4.2 Functionality out of scope 4

2. Solution Overview 5

2.1 Logical View 5

2.1.1 Key Architectural Decisions 5

2.1.2 Subsystems 5

2.1.3 Platform Design 5

2.1.4 Tiers and Layers 5

2.2 Physical View 5

2.2.1 Technology and Services 6

2.2.2 Key Architectural Decisions 6

2.3 Data Design 6

2.4 Other Architectural Decisions 6

2.5 Architectural Issues 7

3. Quality Attributes 8

3.1 Performance 8

3.2 Availability 8

3.3 Security 8

3.4 Maintainability 8

4. DevOps and Deveopment Lifecycle 8

4.1 Source Control Strategy 8

4.2 Continuous Integration 8

4.3 Continuous Delivery 8

# Introduction

## Background

Provide the background and context of the project and its current status from a project perspective.

## Business Needs

Include a brief summary of the business needs and drivers for the initiative.

## Stakeholders

List the key business and IT stakeholders in the initiative.

* Us
* iJooz Company

## Project Scope

Briefly describe the scope of the solution. State what is included in the scope and what is out of scope. If relevant, define the phases of the solution and what is delivered in each phase. Provide references to the documented scope. Also clearly define the scope for the designs and implementations if you do not implement everything that you have designed

**All functionalities that will be needed by the platform (this is not delivery scope)**

### Functionality in scope

Briefly list down the functionality and use cases that you include as part of the project

**All delivery scope**

### Functionality out of scope

Briefly list down the functionality and use cases that is out of scope of the project

**All undelivered project scope**

# Solution Overview

## Logical View

Describe the architecturally significant parts (or elements) of the solution and the design model showing its decomposition into subsystems using logical detail deployment diagram and other diagrams as necessary.

The individual elements of the solution must be clearly defined and assigned to the respective providers. An element (or subset of multiple elements) may be allocated to a Vendor.

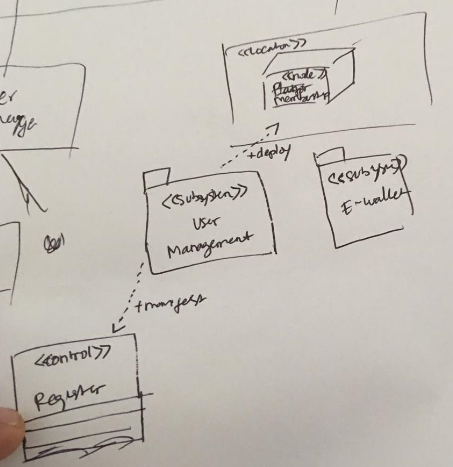
A solution element is any part of the architecture of the overall solution, e.g., a COTS product, a custom-built software module, a data repository, a network device, etc. Each element can utilise the services provided by other elements and provide services of its own.

The logical view describes how elements participate in the solution. It includes static and dynamic relationships and interactions between elements. The documentation typically includes a number of diagrams expressing the different kinds of relationships — for example, dependency relationships, usage relationships, interaction relationships, etc

**Copy and paste the section 2.1 from our project proposal (except subsystem/microservice detail) and align it with current development**

**And add 1 more outline functional elements to show relationship between microservices**

**And also to draw something like this for logical deployment diagram**



### Key Architectural Decisions

Provide a summary of the significant decisions made in arriving at the architecture of the solution that are related and/or reflected in the logical view. Documented decisions should focus on trade-off choices and rationale.

The key architectural decisions taken are as follows:

|  |  |
| --- | --- |
| Identifier | Description |
| AD-01 | *…* |
| AD-02 | *…* |

Why we’re using microservice approach => search in google with keyword “why we use microservice”

Why we’re using EWallet service (not merge it to QR service) => Flexibility to the platform to not just sell QR code in the future

### Subsystems

Describe the architecturally significant parts (or elements) of the solution and the design model showing its decomposition into subsystems.

**Microservice logical architecture detail which is in section 2.3 of project proposal**

### Platform Design

Specifically describe how the platform is designed and interact with the system that leverage the platform (the applications)

Kevin will take care this part

### Tiers and Layers

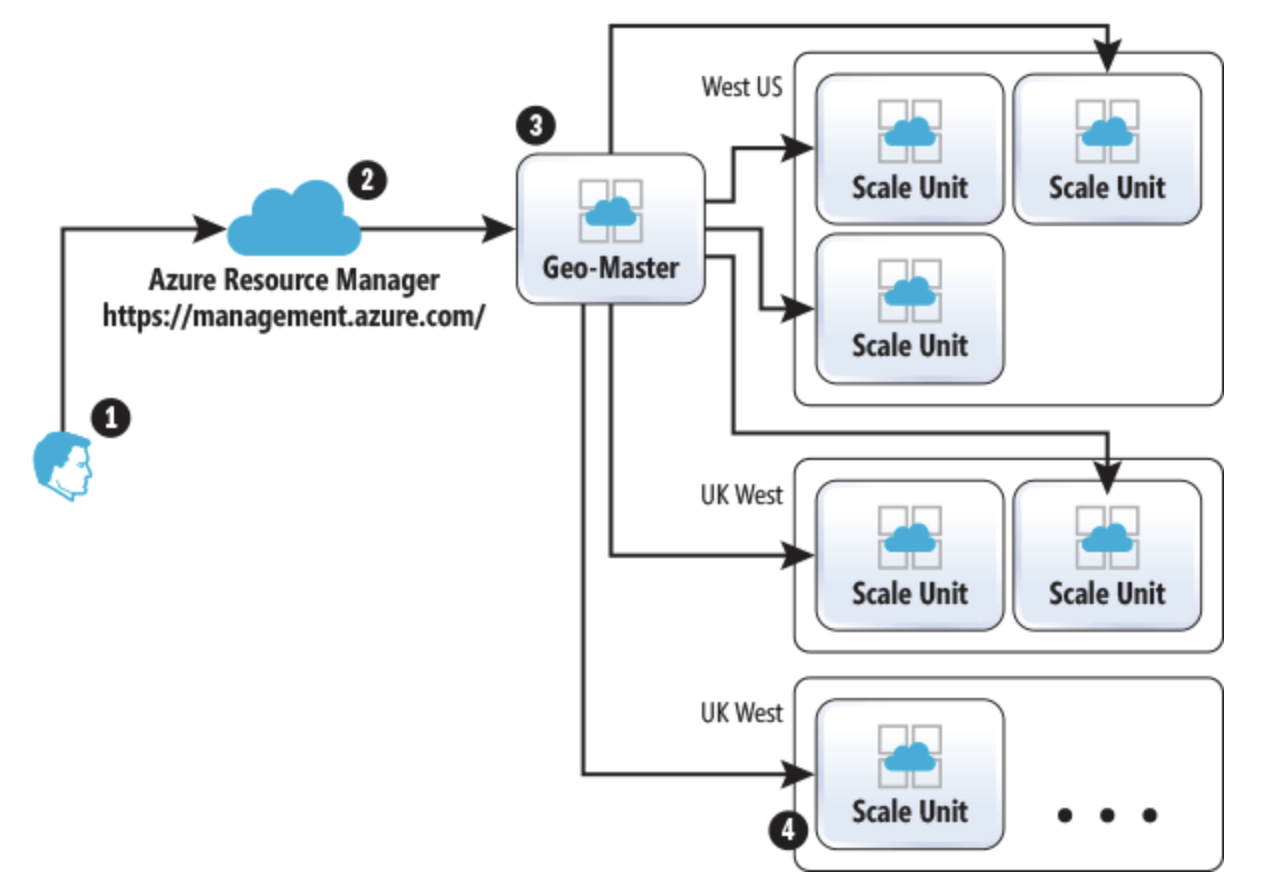
Describe the layers and their responsibility in the system, e.g., Presentation, Business Logic, Data and Integration Tiers. Rename the section is you organized the solutions using different method (not tiered or layered approach)

3 Tiers which are:

* Presentation => Web UI
* Business logic => Microservices
* Data => SQL database

## Physical View

Describe the architecturally significant components (or elements) of the solution by describing the physical design which may include the network, technologies, nodes and machines where the components are deployed using physical detail deployment and other diagram as necessary. This is where you can also include and describe the cloud services that you use if any.



Current Physical diagram

### Technology and Services

Describe the architecturally significant technology and cloud services that is used in the solution and describe the trade-offs and rationale.

* PaaS
* Azure App Service
* Redhat openshift
* .Net Core
* SQL Server
* OAuth2

### Key Architectural Decisions

Provide a summary of the significant decisions made in arriving at the architecture of the solution that are related and/or reflected in the physical view. Documented decisions should focus on trade-off choices and rationale. You should continue the numbering from the previous architectural decisions table

The key architectural decisions taken are as follows:

|  |  |
| --- | --- |
| Identifier | Description |
| AD-03 | *…* |
| AD-04 | *…* |

# Why we use PaaS => 1) Ease of maintenance (e.g. don’t have patch the OS). 2) Scalability features from PaaS vendor

# Why we use azure app service => 1) Customer requirement to have the PaaS on Microsoft Platform. 2) We have $100 credit as NUS student to use Azure platform.

* Why we use redhat openshift => due time restriction, and deployment issue of authentication service

Why we use SQL Server

* 1) Data is structured and predictable. 2) Customer already have SQL server license

Why we use .net core

* 1) Developer and customer are familiar with the framework.

Why we use Oauth2 (will be provided by Zou Xuan)

* 1) This is the most convenient we find so far for microservices architecture

Each of us will contribute if there’s anything else

## Data Design

State how data is managed in the solution, e.g., in a relational database. Describe any special considerations for transaction management, concurrency, etc. Data persistence — indicate which parts of the data model must persist beyond a transaction or session and the period of persistence. Provide a logical data model for the persistent data storage used in the solution.

Describe the information architecture of the solutions describing the type of data inventory and classification that you have and how the data will be stored, secured and handled. E.g.:

<Data Group>

Data format: <explain the different type of data that belong to this group and how the data is stored as record or flat file or other form of data, in what format(JSON, XML, etc)

Proposed storage technology: <explain the technology used to store this data e.g. Oracle RDBMS, MongoDB document database, etc with the rationale>

Security requirements and controls: <explain in high level the security requirement for this data and what are the security controls that are applicable for this data>

Kunal and Kevin will be working on this. Fendy will share the database connection string

Fendy’s database

Server=tcp:ijoozewallet.database.windows.net,1433;Initial Catalog=ijoozewallet;Persist Security Info=False;User ID={your\_username};Password={your\_password};MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;

Trade-off consideration: <document the trade-off consideration that has been considered in making this decision.

## Other Architectural Decisions

Provide a summary of the other significant decisions made in arriving at the architecture of the solution that has not been mentioned. If you explain the rationale of the decisions in other section (such as the quality attribute sections), you can make a reference to those sections.

The key architectural decisions taken are as follows:

|  |  |
| --- | --- |
| Identifier | Description |
| AD-03 | *…* |
| AD-04 | *…* |

Not Available

## Architectural Issues

(Optional but it’s important to acknowledge any known limitation of your solution before any stakeholder point them out to you) Include any outstanding issues from a solution architecture and high-level design perspective which need to be resolved. For resolutions that extend beyond existing project plans, there should be a roadmap with a timeline for closing the issue.

The key outstanding architectural issues in the solution are as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifier | Issue and impact | Description | Resolution | Owner | Status |
| AISS–01 | Identify the issue and its impact | Describe the issue and its impact | Describe the resolution for the issue | Owner of the issue | E.g., Open, In Progress, Closed, etc. |
| AISS–02 | … | … | … | … | … |

# Quality Attributes

## Performance

The purpose of this section is to describe the solution handles the performance requirement of the system. You can describe the use of caching, load balancing, any optimization that you plan/have done in the design of the system

* Automated scalability from cloud provider (azure and redhat openshift)

## Availability

The purpose of this section is to describe how the solution handles high availability requirement.

* Automated scalability from cloud provider (azure and redhat openshift)

## Security

The purpose of this section is to describe the security that will be incorporated into the solution which includes any security services: IAM, encryption, hashing, IDS, IPS, etc.

* **JWT**
* **Password hashing**
* **Firewalls**
* **OAuth2**

## Maintainability

The purpose of this section is to describe the solution handles the maintainability requirement. How the platform and new application can be enhanced and maintained in the future. You can refer to the platform section if necessary.

* **Refer to 07. Ensure Maintainable Architecture file, and pick 3 items**

# DevOps and Deveopment Lifecycle

## Source Control Strategy

Describe the strategy that you use to manage your source code and other artefacts in your configuration management. Include your project structures, repository strategy, branching strategy, authentication of develoeprs, etc.

Each microservice has their own Git repository

## Continuous Integration

Describe the pipeline on how you perform continuous integration in your project including the trigger for each integration jobs, the various jobs, tests, etc.

Each microservice has their own CI

## Continuous Delivery

Describe the pipeline on how you automate deployment to different environment including how many environments are setup, who is responsible for approving promotion between environments and how deployment get verified.

Each microservice has their own CD

# <other things to be highlighted>