# Teilnehmer/innen des Teams:

|  |  |
| --- | --- |
| Klasse:  WUP22 | Team:  Luca Binder, Carl Otto Strömstedt |

# Anforderungsdefinition (Meilenstein A)

|  |  |
| --- | --- |
| Sunrise-Mensa | |
| **Fachlicher Inhalt:**  (Allgemeine Beschreibung) | Skript das jeden Tag des Tagesmenü der Sunrise-Mensa verschickt Die Homepage von Sunrise wird jeden Tag um 8 Uhr Angefragt.  Homepage:    Die Antwort wird nach den einzelnen Menüs durchsucht. Die Menüs werden dann per Mail durch Outlook versendet.  Das Email sollte in einzelnen Menüs strukturiert sein und so aussehen: |
| **MUSS**  **Kriterien:**  (Konkrete Features, die umzusetzen sind) | Ablauf des Scripts:  * Web-Request * Response einlesen * For-Loop bei dem für jedes Menü ein Text erstellt wird * Outlook-Objekt wird definiert * Emailangaben werden konfiguriert * E-Mail wird versendet |

|  |  |
| --- | --- |
| **KANN**  **Kriterien:**  (Konkrete Features, die optional sind) | **Zusatz:**   * Einstellen welche Menüs man haben möchte * Email angeben an die man das Menü versendet möchte |

## Planung (LB1 / LB2)

|  |  |  |  |
| --- | --- | --- | --- |
| *MS* | *Tätigkeit / Abgabe* | *Soll-Datum* | *Ist-Datum* |
| A | ProjektstartTeam BildungWahl / Ausarbeitung der Anforderungsdefinition (Kap. 2) Abnahme Anforderungsdefinition durch Lehrperson | 02.03.23 |  |
| B | Teamaufgabe 1:Abgabe: Lösungsdesign  (Funktionsmodell / GUI / PAP / Struktogramm / UML AD / Storyboard) | 09.03.23 |  |
| B2 | Teamaufgabe 2:Abgabe: Testvorschrift und Testfälle |  |  |
| C | Einzelaufgabe 2:Abgabe Programmcode und DokumentationFachgespräch Projektabnahme | 30.03.23 |  |
| C2 | Einzelaufgabe 4:Abgabe: Ausgefüllter Systemtest |  |  |

# Lösungsdesign (Meilenstein B: Teamaufgabe 1)

Anhand der Analyse wurde folgendes Lösungsdesign entworfen:

## Struktur

Im Folgenden sind die erwarteten Eingaben und Ausgaben beschrieben / dargestellt:

Diagram

Description automatically generated

(Funktionsmodell / GUI)

## Ablauf

Aus Benutzersicht ist folgender Ablauf des Programms zu erwarten:

Diagram

Description automatically generated

(PAP / UML AD / Storyboard)

## Funktionen

Folgende Funktionen sind hier detailliert dokumentiert:

Diagram

Description automatically generated

(Detaillierte Beschreibung der Funktionen / Struktogramm / UML AD)

# Testvorschrift (LB2 Meilenstein B2: Teamaufgabe 2)

Testbeschrieb und vorbereitetes Testprotokoll siehe Dokument   
***M122\_LB2\_Testvorschrift\_MS-B2.docx***

# Testprotokoll (LB2 Meilenstein C2: individuelle Aufgabe 4)

Ausgefülltes Testprotokoll siehe Dokument   
***M122\_LB2\_Testvorschrift\_MS-C2\_Name.docx***

# Systemdokumentation (Meilenstein C: individuelle Aufgabe 3)

Die erstellten Projekt-WPS-Scripts sind hier detailliert abgelegt:

***M122\_ Aufgabe\_3\_IhrName.zip***

## Umfang / Abgrenzung / Änderungen gegenüber Design

Aufgrund unten beschriebener Umstände sind Anpassungen des ursprünglichen Lösungsdesigns gemacht worden:

...

(Umstände / Anpassungen / Veränderungen)

## Funktionalität der Implementation.

Zusätzlich zu der Inline-Dokumentation sind hier folgende Funktionen detailliert beschrieben:

...

(Ausführliche Beschreibung der internen Funktionen   
oder Verweis zum Inline-Kommentar)

# Betriebsdokumentation (Meilenstein C: individuelle Aufgabe 3)

...

1. Ziel und Funktion des Scripts Dieses PowerShell-Script ruft den aktuellen Menüplan von der Ambassador-Restaurant-Website ab und sendet diesen per E-Mail an eine definierte E-Mail-Adresse.
2. Code-Übersicht Der Code besteht aus mehreren Abschnitten, die für das Abrufen des Menüs, die Formatierung des E-Mail-Texts und das Senden der E-Mail verantwortlich sind.
3. Code-Abschnitte

3.1 Variableninitialisierung

bashCopy code

$email\_text = "" $url = "https://www.ambassador-restaurant.ch/de/menuplan/"

Hier wird die Variable **$email\_text** initialisiert, um den Text für die E-Mail zu speichern. **$url** enthält die URL der Ambassador-Restaurant-Website, von der der Menüplan abgerufen wird.

3.2 Abrufen der Webseite

bashCopy code

$response = Invoke-WebRequest -Uri $url

Der Befehl **Invoke-WebRequest** wird verwendet, um die Webseite abzurufen und in der Variable **$response** zu speichern.

3.3 Extrahieren der Menüinformationen

bashCopy code

$menu\_tab = ($response.ParsedHtml.getElementsByTagName('div') | Where {$\_.id -eq "menu-plan-tab1"}) $menu\_item = ($menu\_tab.getElementsByTagName('h2') | Where {$\_.ClassName -eq "menu-title"}).InnerText $menu\_line = ($menu\_tab.getElementsByTagName('span') | Where {$\_.ClassName -eq "menuline"}).InnerText $menu\_text = ($menu\_tab.getElementsByTagName('p') | Where {$\_.ClassName -eq "menu-description"}).InnerText

In diesen Zeilen wird der Menüplan aus dem HTML der Webseite extrahiert. Dabei werden die relevanten Elemente wie Menütitel, Küchenzeile und Menübeschreibung gesucht und in entsprechenden Variablen gespeichert.

3.4 Formatieren des E-Mail-Texts

perlCopy code

for ($i = 0; $i -lt $menu\_line.Length; $i++){ $email\_text += @" ---------------------- Küche: $($menu\_line[$i]) Titel: $($menu\_item[$i]) ---------------------- $($menu\_text[$i]) ---------------------- "@ }

In dieser Schleife wird der E-Mail-Text formatiert, indem die extrahierten Menüinformationen in einer übersichtlichen Struktur zusammengesetzt werden.

3.5 Erstellen und Senden der E-Mail

perlCopy code

$ol = New-Object -ComObject Outlook.Application $mail = $ol.CreateItem(0) $mail.To = "luca.binder.privat@gmail.com" $date = Get-Date -Format "dd/MM/yyyy" $mail.Subject = "Menu from " + $date $mail.Body = $email\_text $mail.Importance = 1 $mail.Send()

In diesem Abschnitt wird ein neues Outlook-Objekt erstellt und ein E-Mail-Objekt mit den formatierten Menüinformationen als Inhalt erzeugt. Die E-Mail wird an die definierte E-Mail-Adresse gesendet, und der Betreff enthält das aktuelle Datum.

1. Anforderungen und Voraussetzungen

4.1 Betriebssystem und PowerShell-Version Dieses Script wurde für die Ausführung in einer Windows-Umgebung entwickelt und erfordert eine PowerShell-Version 5.1 oder höher.

4.2 Outlook-Installation Das Script verwendet die Outlook-Desktopanwendung, um die E-Mail zu erstellen und zu senden. Daher ist es erforderlich, dass die Outlook-Anwendung auf dem Computer installiert ist und korrekt eingerichtet wurde.

1. Anpassungen und Erweiterungen

5.1 Ändern der E-Mail-Adresse Um die E-Mail an eine andere Adresse zu senden, ändern Sie den Wert der Variable **$mail.To**:

perlCopy code

$mail.To = "beispiel@email.com"

5.2 Anpassen der E-Mail-Betreffzeile Um den Betreff der E-Mail anzupassen, ändern Sie den Wert der Variable **$mail.Subject**:

bashCopy code

$mail.Subject = "Neuer Betreff: " + $date

5.3 Erweitern des Scripts um weitere Funktionen Das Script kann nach Bedarf angepasst oder erweitert werden, z. B. durch das Hinzufügen von:

* Cc- oder Bcc-Empfängern: Fügen Sie die Zeilen **$mail.CC = "beispiel\_cc@email.com"** und/oder **$mail.BCC = "beispiel\_bcc@email.com"** ein.
* E-Mail-Anhänge: Fügen Sie die Zeile **$mail.Attachments.Add("Pfad/zur/Datei.ext")** ein.
* HTML-E-Mail-Formatierung: Ändern Sie den Wert der Variable **$mail.BodyFormat** auf **2** (für HTML) und passen Sie den Inhalt der Variable **$email\_text** entsprechend an.

1. Fehlersuche und Fehlerbehebung Sollten beim Ausführen des Scripts Fehler auftreten, überprüfen Sie bitte:

* Die Verbindung zum Internet
* Die korrekte Installation und Konfiguration von Outlook
* Die Verfügbarkeit und Struktur der Ambassador-Restaurant-Website, da das Script auf der aktuellen Struktur basiert und bei Änderungen möglicherweise angepasst werden muss

1. Zusammenfassung Dieses PowerShell-Script dient dazu, den Menüplan des Ambassador-Restaurants von deren Webseite abzurufen und per E-Mail an eine definierte E-Mail-Adresse zu senden. Es nutzt die Outlook-Anwendung, um die E-Mail zu erstellen und zu versenden, und kann nach Bedarf angepasst oder erweitert werden.

## Installations- / Bedienungsanleitung