**UML-ONTWERP & REALISATIE**

|  |  |
| --- | --- |
| **Naam** | Jay Bhageloe |
| **Studentnummer** | 22072039 |
| **Klas** | SE5 |

**OPDRACHT**

Geef hieronder een korte beschrijving van je opdracht (een samenvatting van je productvisie voor het vak H-SE-S2REQS) zodat je docent OPT ook begrijpt met welke opdracht jij bezig bent:

De productvisie richt zich op het oplossen van het probleem van inefficiënte planning voor studenten, zoals ervaren door de klant, Mike Asante, een Software Engineering student aan Hogeschool Leiden. De app die wordt voorgesteld, is bedoeld om tijdverspilling te voorkomen en het missen van deadlines te verminderen. Het stelt gebruikers in staat om informatie in te voeren over lessen, projecten, deadlines en toetsen, en biedt vervolgens herinneringen en countdowns om hen te helpen bij het plannen en organiseren van hun tijd. De geschatte kosten van het ontwikkelen van de app worden berekend op basis van het uurloon van Mike en de potentiële tijdswinst die de app kan opleveren, wat aantoont dat de app zichzelf op korte termijn kan terugverdienen door het verhogen van Mike's productiviteit.

**GITHUB**

Maak je GitHub-repo private en geef je docent voor het practicum toegang tot je repo (vraag hem om zijn account als hij dat volgens jou nog niet aan jou gegeven heeft):

https://github.com/SchoolAccount22072/Project2OPT

**UML-ONTWERP**

Hieronder heb ik een kopie gemaakt van mijn ontwerp. Het is verleidelijk om hier een ingewikkeld model te laten zien, maar dat hoeft niet. Zie het volgende document voor voorbeelduitwerking en aanwijzingen:

[OPT2 - Portfolio - UML-Classmodel en -realisatie - Toelichting.pdf](https://brightspace.hhs.nl/d2l/common/viewFile.d2lfile/Content/638434402350496172/OPT%20-%20Portfolio-opdracht%202%20-%20UML%20Class%20Diagram%20en%20realisatie%20-%20Toelichting.pdf?ou=56744&fid=L2NvbnRlbnQvZW5mb3JjZWQvNTY3NDQtSC1TRS1TMk9QVC0yM18yMDIzX1ZUL2RvY3VtZW50ZW4vT3BkcmFjaHRlbi9PUFQgLSBQb3J0Zm9saW8tb3BkcmFjaHQgMiAtIFVNTCBDbGFzcyBEaWFncmFtIGVuIHJlYWxpc2F0aWUgLSBUb2VsaWNodGluZy5wZGY)

Onderstaand ontwerp heb ik opgesteld om de volgende User Stories te kunnen realiseren. Dat hoeven dus niet alle *User Stories* uit je initiële *backlog* te zijn, maar alleen die *User Stories* waarmee je aan kunt tonen dat je het vak OPT beheerst (gebruik de ‘+’ rechts van de tabel om een User Story toe te voegen):

|  |  |
| --- | --- |
| **ID** | **Omschrijving** |
| US1 | ‘Als eigenaar wil ik een interface waar ik gemakkelijk mijn informatie kan invoeren en kan opslaan en terugzien, zodat ik de deadlines, taken en toetsen terug kan inzien en later kan zien of de vakken, projecten en deadlines zijn voldaan of niet’ |
| US2 | ‘Als eigenaar wil ik specifieke informatie (datum, tijd, vak) kan invoeren voor elke deadline, zodat ik een duidelijk overzicht heb van mijn verplichtingen. |
| US3 | ‘ALS eigenaar WIL IK dat er alleen relevante informatie in de applicatie komt en niet informatie zoals individuele lessen, vrije dagen, bijlessen, etc. ZODAT IK alleen de belangrijke informatie kan zien en niet overbelast wordt met onnodige informatie |
| US4 | Als eigenaar wil ik een overzichtelijke visualisatie van de progressie op basis van gehaald of niet gehaald toetsen en of projecten, zodat ik kan inzien of de applicatie effectief is en of alle taken zoals deadlines zijn voldaan.’. |

A diagram of a function

Description automatically generated with medium confidence

Hieronder som je voor elke **UML** *Class* in je ontwerp op wat de functionaliteit is van die *Class*(voor zover je dat nu al kunt weten) en voor elke associatie in het **UML** *Class Diagram* wat de toegevoegde waarde is van die associatie. Laat expliciet zien welk deel van welke *User Story* je met die *Class* of associatie gaat realiseren en controleer of je hiermee alle functionaliteit kunt realiseren.

|  |  |  |
| --- | --- | --- |
| **Naam van de Class** | **US-ID** | **Functionaliteit van genoemde Classes** |
| Gebruiker | US1 | De gebruiker klasse bestaat als een basis voor een standaard gebruiker wat dan veranderd kan worden op basis van wie het is |
| Student | US4 | De Student klasse overerft van de gebruiker en op basis van de gebruiker voert alle nodige informatie in over wie de gebruiker is |
| Docent | US2 | De Docent klasse overerft van de gebruiker en op basis van de gebruiker voert alle nodige informatie in over wie de gebruiker is |
| Menu | US1 | Het menu zorgt voor een simpele interface waar je gemakkelijk naar alle klasse kan navigeren om informatie te invoeren, opslaan en terugzien. |
| Deadline | US2 | De deadline bestaat als een basis voor alle andere klasse die een datum heeft. Deze deadline klasse wordt gebruikt de deadlines die niet per se een project of toets zijn waar alleen simpele informatie nodig is. |
| Project | US3 | De project klasse overerft van de deadline klasse en neemt alle functies over van de deadline klasse zoals naam en data en gaat verder erover zoals groepsgrootte, projectonderwerp en projectbeschrijving. Dit is handig omdat bij een reguliere deadline entry heb je deze informatie niet nodig maar bij een project wel. |
| Toets | US3 | De Toets klasse overerft van de deadline klasse en neemt alle functies over van de deadline klasse zoals naam en data en gaat er verder over zoals toetstijd, toetsstof, toetslocatie en toetsbeschrijving. Dit is handig omdat bij een reguliere deadline entry heb je deze informatie niet nodig maar bij een toets wel. |
| Daglijst | US4 | De daglijst klasse fungeert als een lijst waar alle entries van de deadline, project en toetsklasse samenkomen en de naam van de entry wordt laten zien en het aantal dagen dat over is gebleven. |
| Rapportage | US4 | De rapportage klasse zorgt als een overview van alle informatie op basis van alles wat er in de vorige klasse zijn ingevuld en laat dit zien in een volledige manier. Op basis van wat er wel en niet is gehaald. |

|  |  |  |
| --- | --- | --- |
| **Naam van de associatie** | **US-ID** | **Toegevoegde waarde van de genoemde associaties** |
| Genereren | US4 | Het menu genereerd een daglijst of rapportage |
| Kiezen | US2 | De Gebruiker (student of docent) kiest een menu optie |
| Aanmaken | US3 | Het menu maakt een nieuw project, toets of deadline waarde aan wat kan worden opgeslagen |
| Opslaan | US1 | In de Invoer sla je informatie op voor de Rapportage en Countdown |

**REALISATIE**

De eerste versie van mijn software die ik heb gebaseerd op het ontwerp:

package org.example;

import java.util.Scanner;

import java.io.IOException;

import java.time.format.DateTimeFormatter;

import java.nio.file.Paths;

import java.time.LocalDate;

import java.time.temporal.ChronoUnit;

import java.util.List;

import static java.nio.file.Files.readAllLines;

import java.nio.file.Files;

import java.nio.file.StandardOpenOption;

public class Main {

public static void main(String[] args) throws IOException {

Scanner scanner = new Scanner(System.in);

Gebruiker currentGebruiker;

System.out.println("Kies gebruiker:");

System.out.println("1. Student (Mike)");

System.out.println("2. Docent (Marcel)");

int userChoice = scanner.nextInt();

scanner.nextLine();

if (userChoice == 1) {

currentGebruiker = new Student();

System.out.println("Welkom Mike!");

} else if (userChoice == 2) {

currentGebruiker = new Docent();

System.out.println("Welkom Marcel!");

} else {

System.out.println("Ongeldige keuze. Probeer opnieuw.");

return;

}

Menu menu = currentGebruiker.createMenu(scanner);

menu.startApp();

}

}

class Gebruiker {

public String naam = "Jay Bhageloe";

public String baan = "Schoonmaak";

public int leeftijd = 20;

public String studienaam = "Informatica";

Menu createMenu(Scanner scanner){

return new Menu(scanner);

}

}

class Student extends Gebruiker {

public Student() {

this.naam = "Mike";

this.baan = "Software Engineer";

this.leeftijd = 21;

this.studienaam = "Software Engineering";

}

}

class Docent extends Gebruiker {

public Docent() {

this.naam = "Marcel";

this.baan = "Docent";

this.leeftijd = 40;

this.studienaam = null;

}

}

class Menu {

public Scanner scanner;

public Menu(Scanner scanner) {

this.scanner = scanner;

}

public void startApp() throws IOException {

System.out.println("Welkom bij de Deadline Reminder Java");

beheerKeuzes();

}

public void beheerKeuzes() throws IOException {

while (true) {

System.out.println("\nKies een optie:");

System.out.println("1. Deadlines");

System.out.println("2. Projecten");

System.out.println("3. Toetsen");

System.out.println("4. Daglijst");

System.out.println("5. Rapportage");

System.out.println("6. Stop de applicatie");

int keuze = scanner.nextInt();

scanner.nextLine();

if (keuze == 6) {

System.out.println("Applicatie wordt gesloten.");

break;

}

switch (keuze) {

case 1:

new Deadline(scanner).voerInfoIn();

break;

case 2:

new Project(scanner).voerInfoIn();

break;

case 3:

new Toets(scanner).voerInfoIn();

break;

case 4:

new Daglijst(scanner).voerInfoIn();

break;

case 5:

new Rapportage(scanner).voerInfoIn();

break;

default:

System.out.println("Ongeldige optie. Probeer opnieuw.");

}

}

}

}

class Deadline {

public String naam;

public LocalDate datum;

public String extraInformatie;

public String bestand = "deadline.csv";

public Scanner scanner;

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd-MM-yyyy");

public Deadline(Scanner scanner) {

this.scanner = scanner;

}

protected void voerNaamIn() {

System.out.println("Voer de deadlinenaam in:");

naam = scanner.nextLine();

}

public void voerInfoIn() throws IOException {

voerNaamIn();

System.out.println("Voer de datum in (dd-MM-yyyy):");

String datumInput = scanner.nextLine();

datum = LocalDate.parse(datumInput, formatter);

System.out.println("Voer extra informatie in:");

extraInformatie = scanner.nextLine();

String nieuweRegel = String.join(",", naam, datum.format(formatter), extraInformatie) + "\n";

Files.write(Paths.get(bestand), nieuweRegel.getBytes(), StandardOpenOption.APPEND, StandardOpenOption.CREATE);

System.out.println(naam + " toegevoegd aan " + bestand);

}

}

class Project extends Deadline {

public int groepsgrootte;

public String projectOnderwerp;

public String projectBeschrijving;

public Project(Scanner scanner) {

super(scanner);

this.bestand = "project.csv";

}

@Override

protected void voerNaamIn() {

System.out.println("Voer de projectnaam in:");

naam = scanner.nextLine();

}

@Override

public void voerInfoIn() throws IOException {

voerNaamIn();

System.out.println("Voer de datum in (dd-MM-yyyy):");

String datumInput = scanner.nextLine();

datum = LocalDate.parse(datumInput, formatter);

System.out.println("Voer extra informatie in:");

extraInformatie = scanner.nextLine();

System.out.println("Voer de groepsgrootte in:");

groepsgrootte = scanner.nextInt();

scanner.nextLine();

System.out.println("Voer het projectonderwerp in:");

projectOnderwerp = scanner.nextLine();

System.out.println("Voer de projectbeschrijving in:");

projectBeschrijving = scanner.nextLine();

String nieuweRegel = String.format("%s,%s,%s,%d,%s,%s",

naam,

datum.format(formatter),

extraInformatie,

groepsgrootte,

projectOnderwerp,

projectBeschrijving) + "\n";

Files.write(Paths.get(bestand), nieuweRegel.getBytes(), StandardOpenOption.APPEND, StandardOpenOption.CREATE);

System.out.println("Projectinformatie toegevoegd aan " + bestand);

}

}

class Toets extends Deadline {

public String toetstijd;

public String toetsstof;

public String toetslocatie;

public String toetsbeschrijving;

public Toets(Scanner scanner) {

super(scanner);

this.bestand = "toets.csv";

}

@Override

protected void voerNaamIn() {

System.out.println("Voer de toetsnaam in:");

naam = scanner.nextLine();

}

@Override

public void voerInfoIn() throws IOException {

voerNaamIn();

System.out.println("Voer de datum in (dd-MM-yyyy):");

String datumInput = scanner.nextLine();

datum = LocalDate.parse(datumInput, formatter);

System.out.println("Voer extra informatie in:");

extraInformatie = scanner.nextLine();

System.out.println("Voer de toetstijd in:");

toetstijd = scanner.nextLine();

System.out.println("Voer de toetsstof in:");

toetsstof = scanner.nextLine();

System.out.println("Voer de toetslocatie in:");

toetslocatie = scanner.nextLine();

System.out.println("Voer de toetsbeschrijving in:");

toetsbeschrijving = scanner.nextLine();

String nieuweRegel = String.format("%s,%s,%s,%s,%s,%s,%s\n", naam, datum.format(formatter), extraInformatie, toetstijd, toetsstof, toetslocatie, toetsbeschrijving);

Files.write(Paths.get(bestand), nieuweRegel.getBytes(), StandardOpenOption.APPEND, StandardOpenOption.CREATE);

System.out.println("Toets informatie toegevoegd aan " + bestand);

}

}

class Daglijst {

private Scanner scanner;

public Daglijst(Scanner scanner) {

this.scanner = scanner;

}

public void voerInfoIn() throws IOException {

String[] files = {"deadline.csv", "project.csv", "toets.csv"};

LocalDate vandaag = LocalDate.now();

for (String file : files) {

try {

List<String> regels = readAllLines(Paths.get(file));

for (String regel : regels) {

String[] delen = regel.split(",");

if (delen.length >= 2) {

String naam = delen[0];

LocalDate datum = LocalDate.parse(delen[1], DateTimeFormatter.ofPattern("dd-MM-yyyy"));

long dagenOver = ChronoUnit.DAYS.between(vandaag, datum);

System.out.printf("%s van %s, aantal dagen over: %d%n", naam, file, dagenOver);

}

}

} catch (IOException e) {

System.out.println("Kon " + file + " niet lezen: " + e.getMessage());

}

}

}

}

class Rapportage extends Deadline {

private Scanner scanner;

private DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd-MM-yyyy");

public Rapportage(Scanner scanner) {

super(scanner);

}

@Override

public void voerInfoIn() throws IOException {

String[] files = {"deadline.csv", "project.csv", "toets.csv"};

for (String file : files) {

System.out.println("\nInhoud van " + file + ":");

try {

List<String> regels = readAllLines(Paths.get(file));

for (String regel : regels) {

String[] parts = regel.split(",");

if (parts.length > 2) {

String naam = parts[0];

String datum = parts[1];

String extraInfo = parts[2];

System.out.printf("%s, Datum: %s, Extra Info: %s%n", naam, datum, extraInfo);

if (file.equals("project.csv") && parts.length > 5) {

System.out.printf("Groepsgrootte: %s, Onderwerp: %s, Beschrijving: %s%n", parts[3], parts[4], parts[5]);

} else if (file.equals("toets.csv") && parts.length > 6) {

System.out.printf("Toetstijd: %s, Toetsstof: %s, Locatie: %s, Beschrijving: %s%n", parts[3], parts[4], parts[5], parts[6]);

}

}

}

} catch (IOException e) {

System.out.println("Kon " + file + " niet lezen: " + e.getMessage());

}

}

}

}