

E24 Restaurant

Younes SAOUDI	Issam HABIBI
Chaimaa LOTFI	Mehdi WISSAD
Hatim MESKINE	Hamza MOUDDENE

Technologies Objet: Projet Long

2019 - 2020



Plan de l'exposé

- 1 Démonstration de l'Application
- 2 Présentation Technique
- 3 Organisation de l'Equipe

Démonstration



E24

Welcome to E24 Restaurant !

Architecture

Paquetage

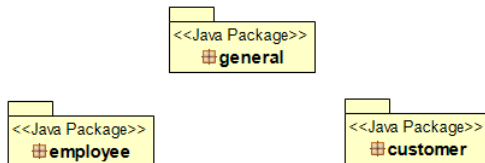


FIGURE – Paquetage de l'Application

Architecture

Diagrammes de Classes

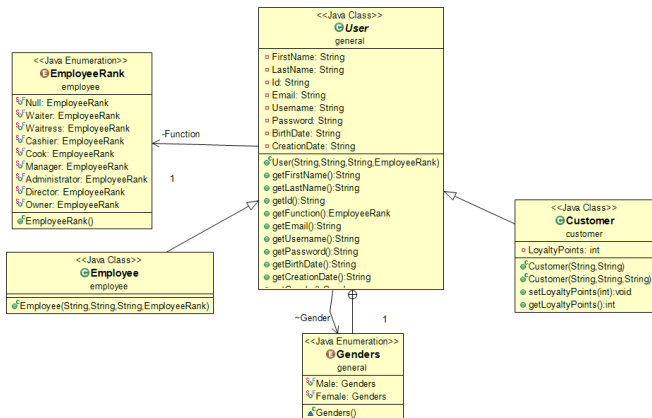


FIGURE – Les Utilisateurs

Architecture

Diagrammes de Classes

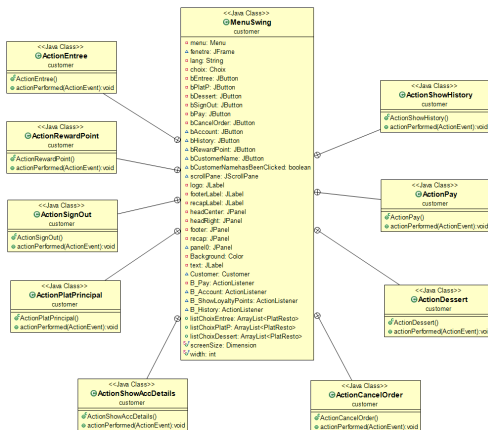


FIGURE – L'Interface Graphique des Clients

Architecture

Diagrammes de Classes

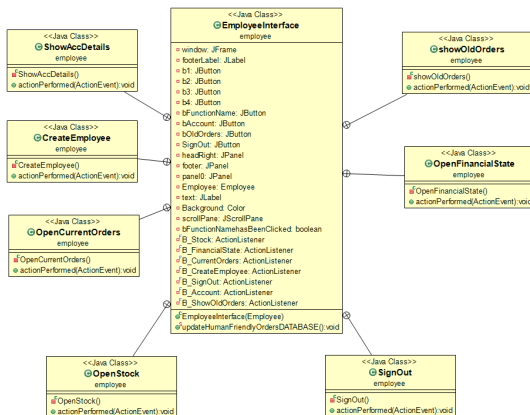


FIGURE – L' Interface Graphique des Employés

Choix de Conception et Réalisation

Observer & HashMap

- **Observer :**
 - Assure la l'interactivité des différentes parties de l'application
 - Offre la possibilité d'ajouter des nouvelles options dynamiques
- **HashMap :**
 - Construit des structures de données et les manipule.
 - Optimise la complexité du travail

Choix de Conception et Réalisation

Observer & HashMap

- **Observer :**
 - Assure la l'interactivité des différentes parties de l'application
 - Offre la possibilité d'ajouter des nouvelles options dynamiques
- **HashMap :**
 - Construit des structures de données et les manipule.
 - Optimise la complexité du travail

Choix de Conception et Réalisation

Observer & HashMap

- **Observer :**
 - Assure la l'interactivité des différentes parties de l'application
 - Offre la possibilité d'ajouter des nouvelles options dynamiques
- **HashMap :**
 - Construit des structures de données et les manipule.
 - Optimise la complexité du travail

Choix de Conception et Réalisation

Observer & HashMap

- **Observer :**
 - Assure la l'interactivité des différentes parties de l'application
 - Offre la possibilité d'ajouter des nouvelles options dynamiques
- **HashMap :**
 - Construit des structures de données et les manipule.
 - Optimise la complexité du travail

Choix de Conception et Réalisation

Observer & HashMap

- **Observer :**
 - Assure la l'interactivité des différentes parties de l'application
 - Offre la possibilité d'ajouter des nouvelles options dynamiques
- **HashMap :**
 - Construit des structures de données et les manipule.
 - Optimise la complexité du travail

Choix de Conception et Réalisation

Observer & HashMap

- **Observer :**
 - Assure la l'interactivité des différentes parties de l'application
 - Offre la possibilité d'ajouter des nouvelles options dynamiques
- **HashMap :**
 - Construit des structures de données et les manipule.
 - Optimise la complexité du travail

Choix de Conception et Réalisation

operations.csv

- Base de données `operations.csv` :
 - Rapide à accéder, facile à manipuler en terme de lecture et d'écriture
 - Garde une sauvegarde du fichier même en cas de panne électrique

Choix de Conception et Réalisation

operations.csv

- Base de données `operations.csv` :
 - Rapide à accéder, facile à manipuler en terme de lecture et d'écriture
 - Garde une sauvegarde du fichier même en cas de panne électrique

Choix de Conception et Réalisation

operations.csv

- Base de données `operations.csv` :
 - Rapide à accéder, facile à manipuler en terme de lecture et d'écriture
 - Garde une sauvegarde du fichier même en cas de panne électrique

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex :orders.json -> Current Orders*)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex :orders.json -> Current Orders*)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex :orders.json -> Current Orders*)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex* : `orders.json` -> `Current Orders`)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex :orders.json -> Current Orders*)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex* : `orders.json` -> `Current Orders`)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex* : `orders.json` -> `Current Orders`)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

JSON et JTable

- Bases de Données JSON `orders.json` et `users.json` :
 - Facile à utiliser, Syntaxe simple et lisible
 - Très Rapide
- Tableaux Swing JTable :
 - Présente les informations de façon concise
 - De JSON à JTable facilement (*ex* : `orders.json` -> `Current Orders`)
 - Dynamique et Modifiable (modifications directement enregistrées sur la base de données grâce à `TableModelListener`)
 - Facilité de trouver l'information grâce au filtrage avec `RowFilters`

Choix de Conception et Réalisation

Cryptage des Mots de Passe et UUID

- Cryptage en base 64 avec un salt (Salage)
 - Augmenter la sécurité des données des utilisateurs qu'ils soient clients ou employés
 - Utilisation des bibliothèques `java.util.Base64` et `javax.crypto`
- Utilisation d'identifiants uniques UUID pour chaque client et chaque commande :
 - Bibliothèque `java.util.UUID`
 - Permet de discerner les utilisateurs et les commandes entre eux ainsi que de vite parcourir les bases de données json

Choix de Conception et Réalisation

Cryptage des Mots de Passe et UUID

- Cryptage en base 64 avec un salt (Salage)
 - Augmenter la sécurité des données des utilisateurs qu'ils soient clients ou employés
 - Utilisation des bibliothèques `java.util.Base64` et `javax.crypto`
- Utilisation d'identifiants uniques UUID pour chaque client et chaque commande :
 - Bibliothèque `java.util.UUID`
 - Permet de discerner les utilisateurs et les commandes entre eux ainsi que de vite parcourir les bases de données json

Choix de Conception et Réalisation

Cryptage des Mots de Passe et UUID

- Cryptage en base 64 avec un salt (Salage)
 - Augmenter la sécurité des données des utilisateurs qu'ils soient clients ou employés
 - Utilisation des bibliothèques `java.util.Base64` et `javax.crypto`
- Utilisation d'identifiants uniques UUID pour chaque client et chaque commande :
 - Bibliothèque `java.util.UUID`
 - Permet de discerner les utilisateurs et les commandes entre eux ainsi que de vite parcourir les bases de données json

Choix de Conception et Réalisation

Cryptage des Mots de Passe et UUID

- Cryptage en base 64 avec un salt (Salage)
 - Augmenter la sécurité des données des utilisateurs qu'ils soient clients ou employés
 - Utilisation des bibliothèques `java.util.Base64` et `javax.crypto`
- Utilisation d'identifiants uniques UUID pour chaque client et chaque commande :
 - Bibliothèque `java.util.UUID`
 - Permet de discerner les utilisateurs et les commandes entre eux ainsi que de vite parcourir les bases de données json

Choix de Conception et Réalisation

Cryptage des Mots de Passe et UUID

- Cryptage en base 64 avec un salt (Salage)
 - Augmenter la sécurité des données des utilisateurs qu'ils soient clients ou employés
 - Utilisation des bibliothèques `java.util.Base64` et `javax.crypto`
- Utilisation d'identifiants uniques UUID pour chaque client et chaque commande :
 - Bibliothèque `java.util.UUID`
 - Permet de discerner les utilisateurs et les commandes entre eux ainsi que de vite parcourir les bases de données json

Choix de Conception et Réalisation

Cryptage des Mots de Passe et UUID

- Cryptage en base 64 avec un salt (Salage)
 - Augmenter la sécurité des données des utilisateurs qu'ils soient clients ou employés
 - Utilisation des bibliothèques `java.util.Base64` et `javax.crypto`
- Utilisation d'identifiants uniques UUID pour chaque client et chaque commande :
 - Bibliothèque `java.util.UUID`
 - Permet de discerner les utilisateurs et les commandes entre eux ainsi que de vite parcourir les bases de données `json`

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Sign Up - Sign In - `users.json` : Hamza Mouddene & Younes Saoudi
- Users Abstract Class: Younes Saoudi
- Customer Interface : Issam Habibi & Younes Saoudi
- Menu: Issam Habibi
- Payment: Hamza Mouddene
- `orders.json`: Younes Saoudi
- Employee Interface : Younes Saoudi
- Stock: Mehdi Wissad & Hamza Mouddene
- `date.csv` - `inventory.csv`: Mehdi Wissad
- Financial State - `operations.csv`: Chaimaa Lotfi

Organisation

Répartition

- Loyalty Points: Hatim Meskine
- Internationalization: Hatim Meskine & Younes Saoudi

Organisation

Répartition

- Loyalty Points: Hatim Meskine
- Internationalization: Hatim Meskine & Younes Saoudi

Organisation

Méthodes Agiles

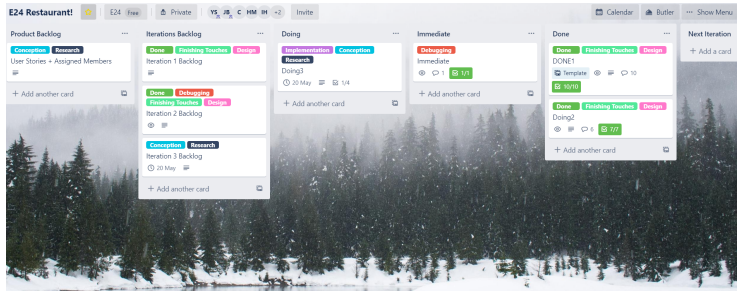


FIGURE – Burndown Chart Trello

Organisation

Méthodes Agiles

- Réalisation de la Burndown Chart :
 - Product Backlog
 - 3 Iteration Backlogs & Doing -> Done -> Debugging

- Réunions (quasi) quotidiennes
- Pair Programming
- Test et Compilation de l'Application d'une façon régulière

Organisation

Méthodes Agiles

- Réalisation de la Burndown Chart :
 - Product Backlog
 - 3 Iteration Backlogs & Doing -> Done -> Debugging
- Réunions (quasi) quotidiennes
- Pair Programming
- Test et Compilation de l'Application d'une façon régulière

Organisation

Méthodes Agiles

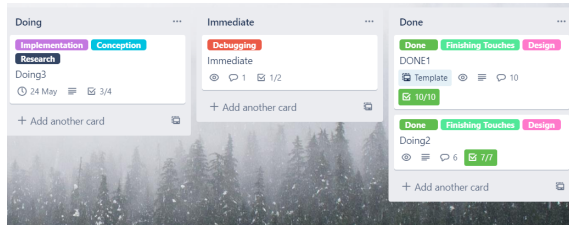
- Réalisation de la Burndown Chart :
 - Product Backlog
 - 3 Iteration Backlogs & Doing -> Done -> Debugging

- Réunions (quasi) quotidiennes
- Pair Programming
- Test et Compilation de l'Application d'une façon régulière

Organisation

Méthodes Agiles

- Réalisation de la Burndown Chart :
 - Product Backlog
 - 3 Iteration Backlogs & Doing -> Done -> Debugging

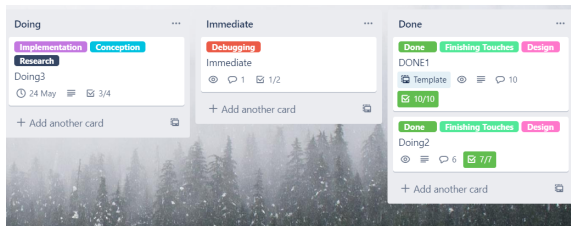


- Réunions (quasi) quotidiennes
- Pair Programming
- Test et Compilation de l'Application d'une façon régulière

Organisation

Méthodes Agiles

- Réalisation de la Burndown Chart :
 - Product Backlog
 - 3 Iteration Backlogs & Doing -> Done -> Debugging

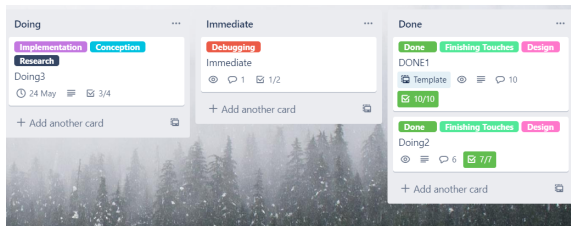


- Réunions (quasi) quotidiennes
- Pair Programming
- Test et Compilation de l'Application d'une façon régulière

Organisation

Méthodes Agiles

- Réalisation de la Burndown Chart :
 - Product Backlog
 - 3 Iteration Backlogs & Doing -> Done -> Debugging

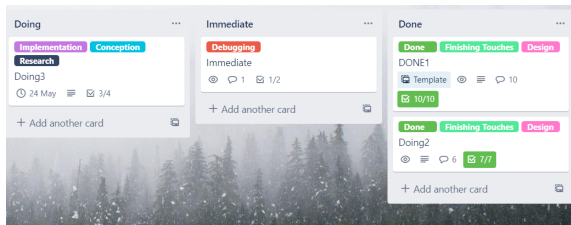


- Réunions (quasi) quotidiennes
- Pair Programming
- Test et Compilation de l'Application d'une façon régulière

Organisation

Méthodes Agiles

- Réalisation de la Burndown Chart :
 - Product Backlog
 - 3 Iteration Backlogs & Doing -> Done -> Debugging



- Réunions (quasi) quotidiennes
- Pair Programming
- Test et Compilation de l'Application d'une façon régulière

MERCI POUR VOTRE ATTENTION.