**CS4125**

**Project Report**

**Food ordering system**

**Dean Kelly(19272898)**

**Jack Busher(19237499)**

**Quinn Painter(19234201)**

**Harneet Singh(18284728)**

**Table of Contents**

**Cover page 1**

**Table of Contents 2**

**Business Scenario 3**

**Software Lifecycle Model 3**

Waterfall Model **4**

V Model **4**

Agile **4**

**Project Plan 5**

Roles **5**

Schedule **5**

Experience **6**

**Requirements**

**Use Cases 7**

Use Case diagrams **7**

Use case descriptions **8**

**Quality Attributes 10**

Extensibility **10**

Performance **10**

**GUI Prototypes 11**

**System Architecture 12**

Programming Language **12**

Framework **12**

MVC & MVT **12**

**Analyst Design 14**

List of candidate objects 14

Entity Relationship Diagram

Conceptual Class Diagram **16**

Sequence Diagram **17**

State Chart Diagram **18**

**References 19**

**Business scenario**

This project entails the design and implementation of a contactless food ordering system that allows customers to order food and drink in a fast food chain without having to walk up to the till, tell the employee their order and then pay there. It allows them to order food directly from a screen in the store.

Customers will be able to avail of a plethora of features such as choosing which language to order in, customising your order, applying a voucher code. The screen will allow people to enter their log in details as to apply discounts. They may be able to redeem a free product or receive a free discount, based on how long its been since their last visit. This is a customer retention scheme which will encourage them to come back more often.

Customers are also able to receive certain discounts on special events or if there are special circumstances admins are able to apply a promotional discount.

Employees will be able to view order history and update stock.

Admins will be able to edit the menu and run promotions, as well as everything that an employee can do.

We were inspired to build a system like this because we all agreed that the machines that are used in the majority of McDonalds chains are user friendly and speed up the process of ordering your food.

By implementing a system like this we hope to be able to process more orders than a typical ordering system by the till. Due to the ability to take more orders we also think that having this system in place will increase the profitability of a company.

**Software Lifecycle Model**

We decided early on that we wanted to use an agile approach. This was after a discussion in which each team member gave their opinion on how we should go about executing this project. We still felt it was necessary to discuss each model in this report, as we wanted to give our justification to why we chose the agile approach.

**Waterfall Model**

This was discussed by our group and we considered using this model as our approach. We came to the conclusion that this wouldn’t be the best approach for us, as we believed that each of us would be working on different parts, and some parts may take longer than others. If we used this model we would be stuck at certain stages of the project. We also agreed that we weren’t going to get the entire project 100% correct on our first run. We were aware that there was going to be some changes to the project down the line.

Graphical user interface, application, website

Description automatically generated

**V-Model**

This was a strong contender for how we were going to approach the project. We all agreed that testing after each phase was a strong point of this approach. It would have allowed us to move through our project one phase at a time, with clear goals and a definitive way to verify our work and move on. It would have been a great way to track progress accurately, however we decided that the disadvantages outweighed the advantages. We believed that the risk of requirements changing later on in the project would negatively impact our ability to stick to this approach.

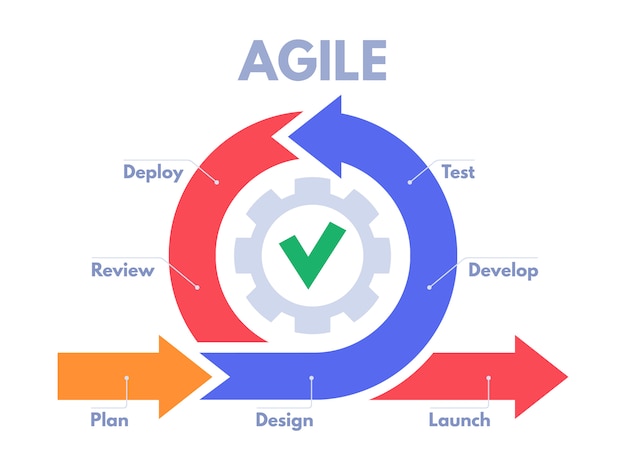
Diagram

Description automatically generated

**Agile**

We ultimately decided that proceeding using the Agile model would be the most beneficial for us. The agile approach would be the most realistic approach in our situation we thought. This was due to a few reasons. One of them was that we wanted to work in teams and we believed that the agile model would promote teamwork and enable it. As mentioned earlier, we believed that we would be making changes to our project along the way and agile seemed like the best way to enable these sort of unforeseen changes. The biggest reason why we chose agile was because we felt that none of us would be holding up the project from advancing any further if we were working on a specific item for a while. With the other approaches, there could have been bottlenecks while one person or a team tries to complete specific task, whereas with this approach we did not feel this problem would be as prominent.

We also discussed the disadvantages of this approach. We knew that the lack of an absolutely strict plan was a strength, it could also lead to being a weakness. We needed to hold each other accountable for what work needed to be done, and to not stray too far from the plan.



**Project Plan**

**Roles**

|  |  |
| --- | --- |
| Project Manager | Jack |
| Document Manager | Dean |
| Business Analyst | Dean/Jack |
| Architect | Quinn/Jack |
| System Analyst | Group |
| Technical Leads | Quinn/Harneet |
| Programmers | Group |
| Tester | Jack/Harneet |
| Dev ops | Quinn |

**Schedule**

|  |  |  |
| --- | --- | --- |
| Task | Assigned | Due |
| Business description | Jack | W3 |
| Software lifecycle | Jack | W4 |
| Project Plan | Dean | W4 |
|  |  |  |
| Use case 1 | Jack | W4 |
| Use case 2 | Dean | W4 |
| Use case 3 | Jack | W5 |
| Quality attributes | Dean | W5 |
| GUI Prototypes | Quinn | W4 |
| MVC Diagram | Harneet | W5 |
| System architecture | Harneet | W6 |
| Candidate objectives | Quinn | W6 |
| Conceptual class diagram | Quinn | W6 |
| Sequence diagram | Jack | W6 |
| State chart | Dean | W6 |
| ERD diagram | Quinn | W6 |
| Code | Group | W12 |
| Testing | Jack/Harneet | W12 |
| Architectural diagram | Dean/Quinn/Jack | W12 |
| Critique | Group | W12 |
| Refernces | Dean | W12 |

**Experience**

**Dean**

I have experience in python and flask. I was able to develop these skills throughout my placement at Ei Electronics.

**Jack**

My placement was at Regeneron. I was mainly working with old databases and converting them to Access databases

**Quinn**

My placement was at Jaguar Land Rover, doing Matlab and Python with QT.

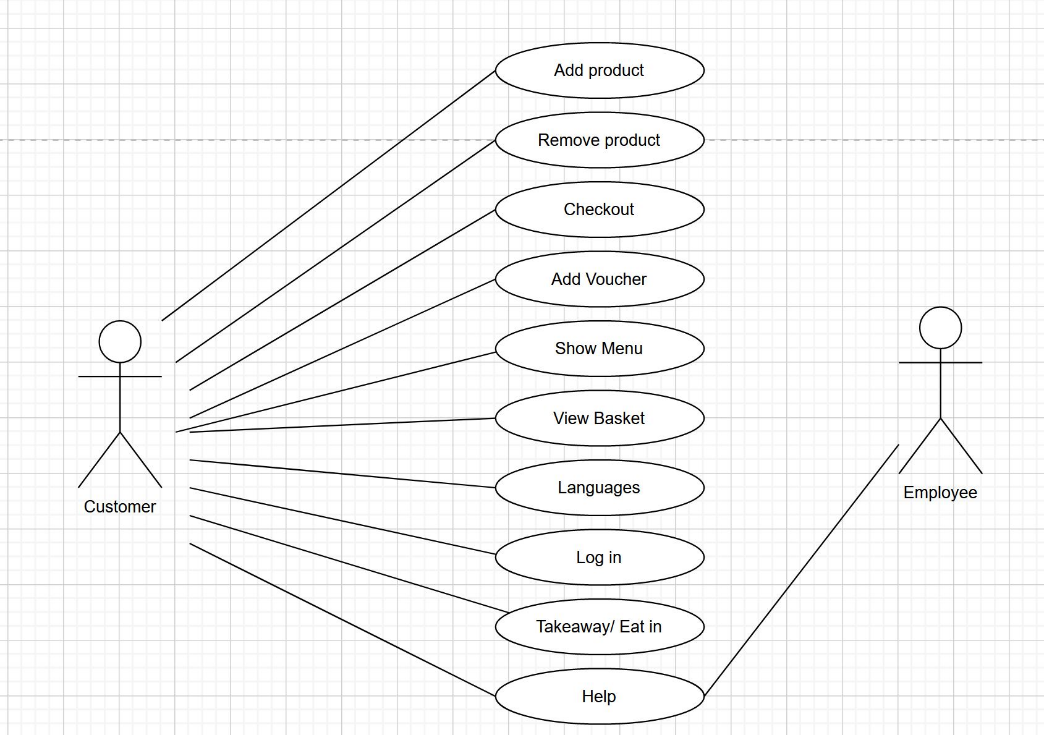
**Harneet**

I have 1 year of work experience in python, my tasks included creating and modifying automated tests scripts while working in Intel, Shannon as part of my coops

My python skills are intermediate.

**Requirements**

**Use Cases**



Diagram

Description automatically generated

|  |  |
| --- | --- |
| USE CASE 1 | ORDERING FOOD |
| Goal in context | The customer wants to view the menu, select the items they want, and pay for the items. |
| Scope & level |  |
| Preconditions | The customer has clicked the screen to activate it and the machine is working |
| Success end conditions | The customer is able to order the food they selected and successfully paid for the food. |
| Failed end conditions | The food isn’t added to the order correctly. |
| Primary, Secondary actors | Customer, employee, admin |
| Trigger |  |
| Description | 1. The customer activates the screen 2. The customer chooses the language 3. The customer chooses whether to enter their ID Number 4. they want to order in 5. Customer selects which items they would like to add to basket 6. Customer selects check out 7. Customer chooses whether to apply a discount code or not 8. The customer chooses take away or eat in 9. The customer pays |
| Extensions | The customer selects an item, but it is out of stock. |
| Variations | The customer chooses to customise their order |
| Related information | Help button |
| Priority | High |
| Performance | Seconds |
| Frequency | 200/day |
| Channel to actors | Not yet determined |
| Open issues |  |
| Due date | Week 8 |
| Superordinates |  |
| Subordinates | Customise order |
|  |  |

|  |  |
| --- | --- |
| USE CASE 2 | Apply a discount |
| Goal in context | Admin is applying a promotional discount/offer |
| Scope & level |  |
| Preconditions | Admin must use their employee login |
| Success end conditions | The system is updated and changes to the menu are applied |
| Failed end conditions | The promotion was never saved and the menu does not reflect the changes made |
| Primary, Secondary actors | Admin |
| Trigger | Admin clicks the apply discount button |
| Description | 1. Admin logs in 2. Clicks apply discount 3. Enters the type of discount (e.g 10% off, print voucher at end of purchase) 4. Enter time period the offer is valid for 5. Confirms the changes. 6. Log off |
| Extensions |  |
| Variations | 1. Discount percentage 2. Time period offer is valid for |
| Related information |  |
| Priority | High |
| Performance | Seconds |
| Frequency | 1 per week |
| Channel to actors | Not yet determined |
| Open issues |  |
| Due date | Week 7 |
| Superordinates |  |
| Subordinates |  |

|  |  |
| --- | --- |
| USE CASE 3 | Pay |
| Goal in context | Successfully pay the correct amount for the food ordered |
| Scope & level |  |
| Preconditions | The customer has added items to the basket |
| Success end conditions | The customer is charged for their food and the company receives the money |
| Failed end conditions | The amount received by the company is wrong, or the amount paid by the customer is wrong.  Payment fails due to insufficient balance |
| Primary, Secondary actors | Customer |
| Trigger | The customer clicks pay |
| Description | 1. The customer clicks checkout when ready to pay for their food 2. This redirects the customer to a checkout page 3. The customer can apply a discount in this page 4. When satisfied with the order price and discounts applied, the customer can select pay 5. This calls an external system which executes a transaction between the customer and the restaurant. |
| Extensions |  |
| Variations | The customer does not add a discount  The customer may choose to pay by cash or card |
| Related information |  |
| Priority | High |
| Performance | Seconds |
| Frequency | 200/day |
| Channel to actors | Not yet determined |
| Open issues |  |
| Due date | Week 10 |
| Superordinates |  |
| Subordinates | Apply discount |
|  |  |

**Quality Attributes**

**Extensibility**

Extensibility represents the ability to add functionality to a component without touching other components or parts of the system. Architectures that involve loose coupling, communication standards and evolution friendly interfaces and schemas promote extensibility.

We will measure how extensible our program is by asking ourselves:

* How fast an account can be made
* If the value of the software is seen almost immediately
* The UI is very user friendly
* The software can be easily expanded on

When designing our system, we prioritized these considerations. We want our customers to spend as little time as possible using the system, to ensure long lines do not build up in store. To do this our systems GUI has to be clear and easy to use.

**Performance**

Performance is described as “the degree to which a system accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage.”

We plan to maximise our performance by:

* Optimising our database and SQL queries
* Ensuring we have only the necessary functions needed to create the food ordering system and receive payment
* Optimizing our UI by adding loading bars and icons to make the experience more manageable.

**GUI PrototypesText

Description automatically generated**

**A picture containing text

Description automatically generated**

**System Architecture**

**Programming Language**

We had a choice between Java and Python and we went with Python as our programming language as most of the team is familiar with it and has used it in their own time and some used it in their coops.

**Framework**

Since our language of choice was python we had two popular frameworks to choose from Django and Flask, they are both written in python and are open source. After doing our research, we decided to go with the Django framework rather than flask because Django supports MVT(Model-View-Template) pattern which is slightly different from the MVC(Model-View-Controller) pattern. On the other hand, the flask doesn't support the MVC pattern(1) and would require us to design it first which in turn would add more time to development cycle.

A quick explanation of how MVT in Django works. In MVC we have a model, view and controller whereas, in Django MVT, we have a model, view and template. The controller part which handles interactions between a model and view is all handled by Django itself. Also slightly confusing part between these two patterns Is how they handle view and controllers. In Below Diagram you can see that View in MVT is not the same as View from MVC because the View in MVC handles how the page is presented to the user but in MVT it’s the template that handles that part and View in MVT is not entirely same as controller from MVC but it act as one to some degree but most of the controller part is handled by Django itself.

Diagram

Description automatically generated

**MVC and MVT difference diagram**

Diagram

Description automatically generated

**Django MVT pattern Diagram.**

Another few reasons we went with Django and not flask, is because Django has some important features that are supplied out of the box with it like -

ORM

-comes with ORM(Object-relational-mapping) that supports many relational databases such as Oracle, MySQL, SQLite etc and we needed MySql.

-Admin

-It’s one of the most powerful thing that comes with Django, it’s a functional admin panel. It will read metadata from our models and provide us with a quick, model-centric interface.

-Authentication

Django comes with a user authentication system that handles authorization and authentication of user accounts, groups, and permissions. Nowadays most if not all apps, and webpages require some form of authentication and authorization.

Database

For the database, we chose one of the popular databases, which is MySQL(Open-Source). Another reason is that most of the team members have some experience with it from oneof our previous semesters (Database Systems) and that would reduce a big chunk of the learning curve. Also as stated above it is fully supported by Django.

**Analyst Design**

**List of candidate objects**

A restaurant has many products for sale. Customers can add several of these products to their order, then check out. Customers can also enter vouchers to reduce the price of the order. When checking out, they can select which payment method to use. Employees can view the order history and update the stock of items. Admins can add, remove and edit products from the menu, and create promotions where voucher codes are sent to customers.

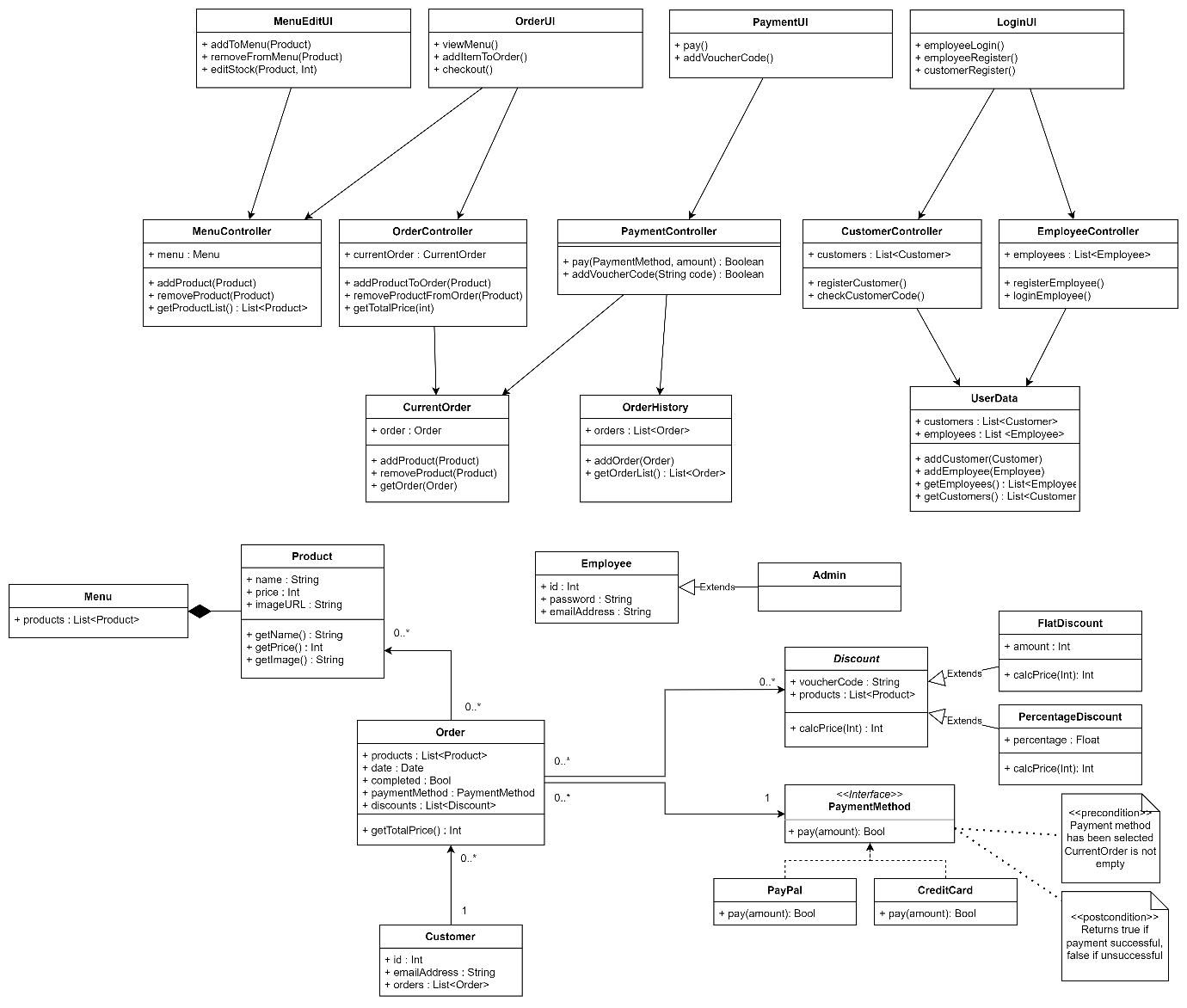
|  |  |
| --- | --- |
| Before | After |
| ~~Restaurant~~ – Outside of scope  ~~Item~~ – Same as Product  ~~Promotion~~ - Event  ~~VoucherCode~~ – Attribute  ~~OrderHistory~~ – Covered by Order  ~~Stock~~ - Attribute  Product  Customer  Order  Voucher  PaymentMethod  Employee  Admin  Menu | Product  Customer  Order  Voucher  PaymentMethod  Employee  Admin  Menu |

**Entity Relationship Diagram**

**Graphical user interface, application

Description automatically generated**

**Conceptual Class Diagram**

****

**Sequence Diagram**

Diagram, schematic

Description automatically generated

**State Chart Diagram**

**Diagram

Description automatically generated**

**References**

https://ieeexplore.ieee.org/document/8901656

V Model

Multi-Criteria Optimization of System Integration Testing - Scientific Figure on ResearchGate. Available from: <https://www.researchgate.net/figure/The-V-Model-for-the-software-development-life-cycle_fig1_329698890> [accessed 17 Oct, 2022]

Agile

Designing and Implementing e-School Systems: An Information Systems Approach to School Management of a Community College in Northern Mindanao, Philippines - Scientific Figure on ResearchGate. Available from: <https://www.researchgate.net/figure/Agile-Methodology-in-System-Development-source-Okeke2021-retrieved-from_fig1_354310848> [accessed 17 Oct, 2022]