**YellowGate Bakery**

**Stakeholders (week 1)**

**Stakeholders: (+ their requirements)**

* Customer
  + Create an order and pay (F)
  + Cancel order
  + Receive their order
  + Able to contact bakery if order is wrong
  + Easy to use (F)
  + Make an account (F)
  + Review order?
* Employees/Team Members/product team
  + Able to read orders
  + Edit order list – remove order when it is completed
  + Edit order menu – remove item if there is no more
* Chief executive
  + Number of customers seeing app
  + Item statistics
  + Customer reviews
  + Sales report
* IT department
  + Low maintenance
* Manager
* Developers
* Company owners
  + Cost of system (N)
  + Sales report
* Extras:
  + Suppliers
  + Delivery drivers

**Requirements (week 2)**

**Functional Requirements: (What systems needs to do)**

* Take orders – create an order and check if it goes through to the employees
* Take payments – Input a payment and check if it takes the payment securely
* Check if order destination is within range
* Find route to destination
* Update/edit Menu e.g., if item is out of stock
* Items needs to have ingredients and allergies written
* Accounts?

**Non-Functional Requirements: (How system should do it)**

* Take payments securely
* Real time transactions of items
* Account? – password
* User guide – novice customers/employees
* Simple design and layout
* Queue of orders so (First In First Out)

**Finding requirements:**

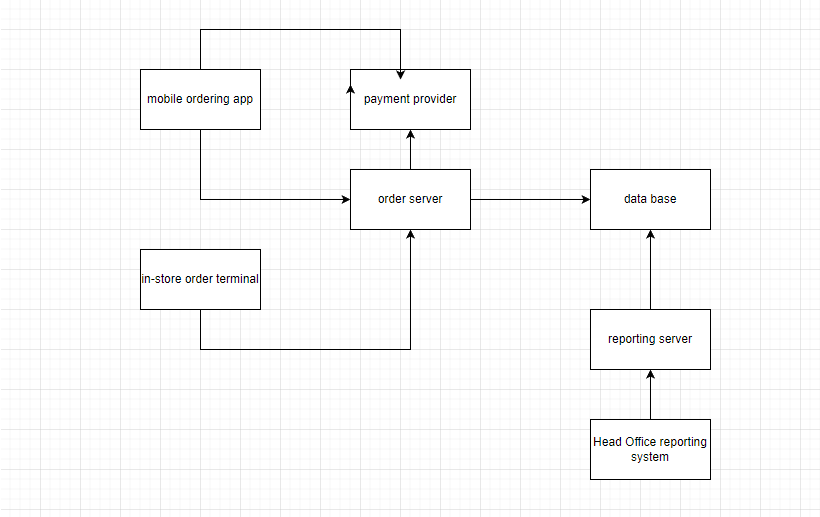
* Group interviews:
  + Staff
* Surveys:
  + Customers
* Document review:
  + Government

**Prioritisation:**

* **Must have:**
  + Be able to order
  + Take payment
  + Refund order
  + Secure
* **Should haves**
  + Items on the app/ descriptions/images
* **Could haves**
  + Loyalty scheme
  + Reviews
  + Reorder
  + Accessibility options
  + Order history
* **Wont haves:**
  + Live tracking
  + GPS

**Design (week 3)**

**Design**



**Testing, Deployment Maintenance (week 4)**

**priority areas:**

App – medium priority – will just crash, user can restart it – error guessing, user acceptance testing, stress testing

Server – high priority – is on public internet, anyone can hack – fuzzing, stress testing

Admin software – low priority – user acceptance testing

* Black box (outside user without knowledge of code tests system)
* White box (developer tests system)
* Manual (user giving input checking output)
* Automated (software running tests and logging results)
* Boundary value analysis (identify boundaries)
  + Mobile ordering app
* Error guessing (identify problematic input values)
  + Mobile ordering app
* User acceptance testing (actual user uses system for intended purpose)
  + Mobile ordering app
* Unit testing (checks unit is working)
* Load/stress testing (test system with huge amount of data)
* Fuzzing (test system with randomly generates inputs)

**Implementation (week 5)**

Repositories: Git

Github – Microsoft, azure.

Will people be able to access the repository from where they are working?

People should be able to access the repository but only when they are working to prevent the risk of third parties getting sensitive data. To do this, the repository could be stored in a server that’s accessed only within the work building.

**Need to consider the risks:**

* Login credentials
* Server addresses
* Encryption keys
* Secure payment transaction
* Data handling

**Pair programming:**

Would the developers likely be working on the same part of the system?

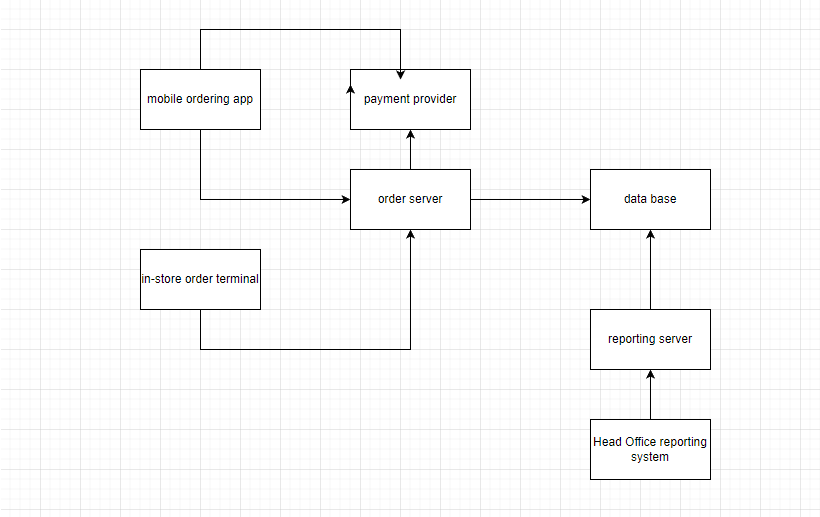
Every developer has a different set of skills and different parts of the system would require different skill levels so it is less likely they will all be working on the same part.

Developers are in the document – pair them together

Ahmed + Alex

**Project Management (week 6)**

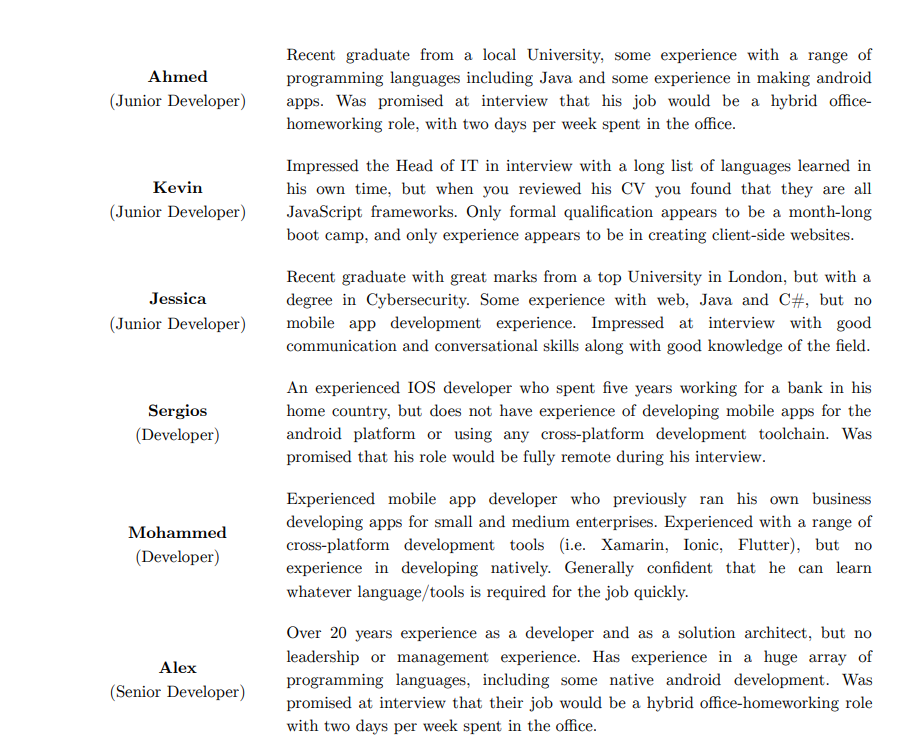
**Task List:**



* Mobile ordering app
  + Design and create visuals for mobile ordering app
  + Implement the mobile ordering app functions
* Payment Provider
  + Create a secure system for the card details to be entered
* Order server

**Dependencies & Critical Path Analysis**

1. Order server – need to create design to know requirement for database
2. Database – to store all data entered from order server
3. In store order terminal – needs to be fully functional
4. Payment provider – needs to be secure payment for customers
5. Mobile ordering app
6. Reporting server
7. Head office reporting system



1. Creating database – 2 weeks – Alex has most experience – database is the most important, so it needs to be done correctly.
2. Mobile ordering app – 2 months – Alex + Ahmed (back end)

Designing visual of the app – 1 week – Kevin

Reporting Servers – 2 months - Jessica + Mohammed + Kevin (once his task is done) + Sergios

1. App Development – 6 months - Mohammed + Kevin + Sergios (front end)

Install kiosk – 3 months - Alex + Jessica + Ahmed

1. End to end testing – Ahmed + Jessica + Sergios + Mohammed

Install computers and software and train staff – Kevin + Alex

**Agile Methodologies - Scrum File (week 7)**

**Scrum**

Pro – suitable for the team size, might be good with changes in plan

Con – inexperience developers,

**User stories**

User:

I want to see what food is available - to see what I want for lunch

Company director:

I want to see the sales report – to see how well the company is doing

**Roles:**

Product owner:

Alex because he is experienced but we want him on the developer team

Mohammed because is also experienced but we want him on scrum

Sergios because he is semi-experienced but he’s fully remote

Scrum master:

Alex has lots of experienced but has no manager experience

Mohammed has lots of experience along with manager experience

Sprint lengths: 4 weeks

First sprint: user, product list, add new product to product list

**Agile Methodologies – Kanban (week 8)**

**Kanban**

Short Kanban boards with small process

Long Kanban (Trello) with long process

Physical Kanban boards are not suitable as people are working from home

Colour cards by priority, or type of job.

Kanban columns should flow from high to low priority.