|  |  |
| --- | --- |
| Team 1 – Lifecycle Justification  and Process Plan | Ivan, Natalia, Ronan  Software Design with AI for Cloud Computing 3.1 |

Contents

[Justification 2](#_Toc152704225)

[Process Plan 3](#_Toc152704226)

[High Level Process Plan 5](#_Toc152704227)

[Database Design 6](#_Toc152704228)

[Architecture 7](#_Toc152704229)

[High Level Components 7](#_Toc152704230)

[Flow Chart 8](#_Toc152704231)

[GUI Architecture 9](#_Toc152704232)

[Sequence Diagram 10](#_Toc152704233)

[Newsagent Sequence Diagram 10](#_Toc152704234)

[Driver Sequence Diagram 11](#_Toc152704235)

[Incremental Delivery Flow 12](#_Toc152704236)

[Activity Bar Chart 12](#_Toc152704237)

[Increment 1: Design Tests / Create Database / Implementation 13](#_Toc152704238)

[Increment 2: Design Tests / Create Database / Implementation 14](#_Toc152704239)

[Increment 3: Link / Integrate / Test Application 15](#_Toc152704240)

[Increment 4: Develop Instruction / Presentation 15](#_Toc152704241)

[Known Defects/Bugs 15](#_Toc152704242)

[NEWSAGENT 15](#_Toc152704243)

[ADMIN 15](#_Toc152704244)

[DRIVER 15](#_Toc152704245)

[UNIT TESTS 15](#_Toc152704246)

# Justification

Our team decided to go for Incremental Delivery, reasons why:

1. **Requirements are well defined** - this allows us to divide the project into smaller phases which we will deliver in increments. Our plan was to pick key increments that would be presented to customer so we can gain valuable feedback and ensure needs are met. After further consideration we decided to involve customer all the way through and present each increment to receive valuable feedback and/or ensure needs are met.
2. **Feedback** - as we want to include customer input/feedback after each increment (which will be highlighted in the plan below), incremental delivery will allow us to present work done to customer. This will also ensure we can incorporate changes at earlier stage if needed.
3. **Supports changes to some extent** – Incremental Delivery is a Plan Driven approach that assumes nothing should change. Although, it doesn’t mean the scope cannot change between increments when required. In the lifecycle we choose, the increments are planned in advance although we still have benefit of making small tweaks if necessary.
4. **Development Process** - it is more visible how much has been done and what’s left to do as each increment is completed and delivered. As we will choose key phases that customer will be involved in, it will be reassuring for them.
5. **Phased Development** – development is done sequentially, where each increment is building up on the previous one. This also doesn’t mean we must cross out parallel development. We can have two developers working on two different increments simultaneously. This will ensure deadline is met and if we are ahead of time, we will have either more time for testing or supporting another developer.
6. **Product Availability** – Incremental Delivery allows us to have working software quicker, which can provide value to customer sooner. It for sure will not be a fully working product after first delivery, although some functionality can be available to customer easier (eg. signing up new users into the system)
7. **Testing** – testing is involved with each increment, which ensures high quality of software is delivered to customer. We also are not moving to the next increment until everything is fully working.
8. **Self-organizing** – we are a self-organizing team which means we are not waiting for a manager (or Scrum) to get the work assigned. We manage our responsibilities and timelines ourselves.

We have decided not to go for Agile as requirements are well-defined and we don’t need to involv4e customer constantly. Instead, we will present what we have at its key phases. We also rejected Waterfall due to its total lack of flexibility. It is highly sequential model, where each phase is dependent on the previous one. We wouldn’t be able to have two devs working on separate phases as everything happens one after another. Here, we also don’t have customer input until the end which can potentially make it very costly to revise/fix if there are issues. We also rejected V-Model as the team is too small to take that approach. It requires extensive testing and parallel working which we don’t have resources for. It would potentially add extra complexity to the project.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Increment | Task | Description | Assignee | Estimation | Actual | Week | Done Yes/No | Comments |
| 1. Initial Planning 2. Specification 3. Design | Team Setup | Prepare working agreement, create/clone team repo | Natalia Ronan Ivan | 4h x each |  | WK 3 | Badge Tick1 with solid fill |  |
| User Stories | Admin, Newsagent, Customer | Natalia | 3h |  | WK 3-4 | Badge Tick1 with solid fill |  |
| Delivery Driver, Publication (book) | Ronan | 2h |  | WK 3-4 | Badge Tick1 with solid fill |  |
| Invoice, Order | Ivan | 2h | 3h | WK 3-4 | Badge Tick1 with solid fill | Underestimated |
| Design System Plan | Design UML for User Stories | Natalia | 2h | 1h | WK 5 | Badge Tick1 with solid fill | Overestimated |
| Ronan | 2h | 3h | WK 5 | Badge Tick1 with solid fill |  |
| Ivan | 2h | 1h | WK 5 | Badge Tick1 with solid fill | Overestimated |
| System Architecture | Describe system architecture | Natalia | 2h |  | WK 4 | Badge Tick1 with solid fill |  |
| Create flow chart | Ivan | 1h | 2h | WK 4 | Badge Tick1 with solid fill | Lack of depth understanding , challenging |
| Create sequence diagram | Ronan | 3h |  | WK 5 | Badge Tick1 with solid fill |  |

# Process Plan

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Increment | Task | Description | Assignee | Estimation | Actual | Week | Done Yes/No | Comments |
| Increment 1 | 1. Design Test Cases (JUnit + MySQL) 2. Create Database 3. Implementation | Customer/ User/Admin | Natalia | 5h | 6h | WK 6-7 | Badge Tick1 with solid fill | Underestimated |
| Publication | Ronan | 5h |  | WK 6-7 | Badge Tick1 with solid fill | Delayed due to internship interviews |
| Order | Ivan | 5h |  | WK 6-7 | Badge Tick1 with solid fill | Delayed due to sickness and internship interview |
|  |  |  |  |  |  |  |  |  |
| Increment 2 | 1. Design Test Cases   (JUnit + MySQL)   1. Create Database 2. Implementation | Newsagent | Natalia | 5h | 3h | WK 8-9 | Badge Tick1 with solid fill | Having Admin/Cust classes done speeded it up as many similarities |
| Delivery Doc | Ronan | 5h |  | WK 8-9 | Badge Tick1 with solid fill |  |
| Invoice, Driver | Ivan | 5h |  | WK 8-9 | Badge Tick1 with solid fill | Driver delayed, Publication will be finished by Sunday |
|  |  |  |  |  |  |  |  |  |
| Increment 3 | Link, Integrate and Test all classes together. App should be fully functioning | Work on CMD | Natalia | 5h | 2h | WK 10-12 | Badge Tick1 with solid fill | Overestimated, creating CMD was much simpler than expected |
| Work on CMD | Ivan | 5h |  | WK 10-12 | Badge Tick1 with solid fill | Will complete Driver on WK12 |
| Work on CMD | Ronan | 5h |  | WK 10-12 | Badge Tick1 with solid fill |  |
|  |  |  |  |  |  |  |  |  |
| Increment 4 | Customer Presentation | Presentation of fully working software to customer | Natalia | 1h |  | WK 13 | TBC |  |
|  |  |  |  |  |  |  |  |  |

## High Level Process Plan

Increment 1: Initial Planning, Specification and Design

* Gather user requirements
* Develop User Stories based on gathered requirements

Entities:

* + Newsagent (6 user stories)
  + Delivery Driver (4 user stories)
  + Admin (6 user stories)
  + Invoices (4 user stories)
  + Publication (2 user stories)
  + Subscription (3 user stories)
  + Delivery Doc (4 user stories)

Total: 29 User Stories (approx.)

* System Architecture
  + Description for system architecture

*Increment 2: Design Tests, Create Databases, Implementation*

* User
* Admin
* Customer
* Publication
* Invoice

*Increment 3: Design Tests, Create Databases, Implementation*

* Newsagent
* Driver
* Delivery Doc
* Order

*Increment 4: Link and Integrate classes, create GUI/CMD*

* Create GUI/CMD flow

## Database Design

* CustomerDetails
  + custID (Primary Key) INT AUTOINCREMENT
  + firstName VARCHAR (10)
  + lastName VARCHAR (10)
  + custAddress VARCHAR (15)
  + phoneNo VARCHAR (13)
* UserDetails (newsagent/driver/admin)
  + userID (Primary Key) INT AUTOINCREMENT
  + username VARCHAR (10)
  + password VARCHAR (255)
  + role ENUM (“admin”, “newsagent”, “driver”)
* DeliveryDriver
  + driverID INT AUTO\_INCREMENT PRIMARY KEY,
  + userFK INT,
  + deliveryDistrict VARCHAR(15)
  + FOREIGN KEY (userFK) REFERENCS userdetails (userID)
* DeliveryDoc
  + docID (Primary Key) INT AUTOINCREMENT
  + custID INT
  + publicationID INT
  + dateDelivered DATE
  + FOREIGN KEY (custID) references Customer\_Details(custID)
  + FOREIGN KEY (publicationID) references Publication(publicationID)
* Invoices
  + invoiceID (Primary Key) INT AUTOINCREMENT
  + custID INT
  + publicationID INT
  + dateDelivered DATE
  + totalSum DECIMAL
  + invoiceDate DATE
  + FOREIGN KEY (custID) references CustomerDetails(custID)
  + FOREIGN KEY (publicationID) references Publication(publicationID)
  + FOREIGN KEY (docID) references DeliveryDoc(docID)
* Publication
  + publicationID (Primary Key) INT AUTOINCREMENT
  + title VARCHAR
  + price DECIMAL
  + quantity INT
  + docID INT
  + FOREIGN KEY (docID) references DeliveryDoc(docID)
* Subscription
  + Relationship -> Customer Details -> Retrieve all details
  + custID
  + subscriptionType VARCHAR
  + custStatus VARCHAR
  + deliveryDisctrict INT
  + startDate DATE
  + endDate DATE
  + FOREIGN KEY (custID) references CustomerDetails(custID)

# Architecture

## High Level Components

1. Newsagent App (Command-Line UI) represents the command line interface through which user interacts with the system. Users input commands to perform various actions, like managing customer details, processing invoices etc.
2. Java Application (Business Logic) contains the logic of the Newsagent application. It interprets user commands, interacts with the MySQL database and performs operations like data retrieval, processing, updating etc.
3. MySQL Database (Data Storage) is where the database tables is stored. The database contains tables for managing customer details, newsagent information, delivery drivers, delivery docs, invoices, publications and subscriptions. Foreign keys establish relationships between the tables for data consistency.

A diagram of a data flow

Description automatically generated

## Flow Chart

A diagram of software

Description automatically generated

GUI ArchitectureA screenshot of a computer screen

Description automatically generated

## Sequence Diagram

**NOTE:**

Sequence Diagrams are also done individually for Newsagent, Order, Publication, Driver, Docket, Admin, and Invoice. To view the full file, please go to Newsagent-Project\Architecture and open “sequence\_diagram.drawio” file in the <https://www.drawio.com/> website.

### Newsagent Sequence Diagram

A diagram of a computer

Description automatically generated

### Driver Sequence Diagram

A diagram of a project

Description automatically generated with medium confidence

# Incremental Delivery Flow

A diagram of a delivery process

Description automatically generated

# Activity Bar Chart

Chart shows the flow of tasks, and dependencies. Eg, T5 is dependent on task T1 and T5 cannot be conducted until T1 is finished.

A screenshot of a computer program

Description automatically generated

### Increment 1: Design Tests / Create Database / Implementation

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Assignee | Description | Completed |
| Increment 1 : WK4-5 | | | |
| # T1 | Natalia | Design test cases for JUnit and sql queries for customer/user/ admin | Badge Tick1 with solid fill |
| # T2 | Ronan | Design test cases for JUnit and MySQL queries for publication | Badge Tick1 with solid fill |
| # T3 | Ivan | Design test cases for JUnit and MySQL queries for invoice | Badge Tick1 with solid fill |
| # T4 | Natalia | Review Ivan’s test cases (invoice) | Badge Tick1 with solid fill |
| # T5 | Ronan | Review Natalia’s test cases (cust, admin, user) | Badge Tick1 with solid fill |
| # T6 | Ivan | Review Ronan’s test cases (publication) | Badge Tick1 with solid fill |
| # T7 | Natalia | Design tables for customer/users | Badge Tick1 with solid fill |
| # T8 | Ronan | Design tables for publication | Badge Tick1 with solid fill |
| # T9 | Ivan | Design tables for invoice | Badge Tick1 with solid fill |
| # T10 | Natalia | Review Ronan’s tables | Badge Tick1 with solid fill |
| # T11 | Ronan | Review Ivan’s tables | Badge Tick1 with solid fill |
| # T12 | Ivan | Review Natalia’s tables | Badge Tick1 with solid fill |
| # T13 | Natalia | Test mysql queries for created tables | Badge Tick1 with solid fill |
| # T14 | Ronan | Test mysql queries for created tables | Badge Tick1 with solid fill |
| # T15 | Ivan | Test mysql queries for created tables | Badge Tick1 with solid fill |
| # T16 | Natalia | Create DatabaseManager class/test | Badge Tick1 with solid fill |
| # T17 | Natalia | Create User/Admin class | Badge Tick1 with solid fill |
| # T18 | Natalia | Create Customer class | Badge Tick1 with solid fill |
| # T19 | Ronan | Review User/Admin class | Badge Tick1 with solid fill |
| # T20 | Ivan | Review Database class | Badge Tick1 with solid fill |
| # T21 | Ivan | Review Customer class | Badge Tick1 with solid fill |
| # T22 | Ronan | Create Publication class | Badge Tick1 with solid fill |
| # T23 | Ivan | Review Publication class | Badge Tick1 with solid fill |
| # T24 | Ivan | Create Invoices class | Badge Tick1 with solid fill |
| # T25 | Ronan | Review Invoices class | Badge Tick1 with solid fill |

### Increment 2: Design Tests / Create Database / Implementation

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Assignee | Description | Completed |
| Increment 2: WK 8-9 | | | |
| # T26 | Natalia | Design test cases for JUnit and MySQL queries for newsagent | Badge Tick1 with solid fill |
| # T28 | Ivan | Design test cases for JUnit and MySQL queries for order/driver | Badge Tick1 with solid fill |
| # T30 | Ronan | Review Ivan’s test cases (order/driver) | Badge Tick1 with solid fill |
| # T31 | Ivan | Review Natalia’s test cases (newsagent) | Badge Tick1 with solid fill |
| # T32 | Natalia | Create Newsagent class | Badge Tick1 with solid fill |
| # T33 | Ronan | Create delivery doc database | Badge Tick1 with solid fill |
| # T34 | Ivan | Create order database | Badge Tick1 with solid fill |
| # T35 | Natalia | Review Ivan’s database | Badge Tick1 with solid fill |
| # T36 | Ronan | Review Newsagent class | Badge Tick1 with solid fill |
| # T37 | Ivan | Review Ronan’s database | Badge Tick1 with solid fill |
| # T38 | Ivan | Create Driver class | Badge Tick1 with solid fill |
| # T39 | Ronan | Create Delivery Doc class | Badge Tick1 with solid fill |
| # T40 | Ivan | Create Order class | Badge Tick1 with solid fill |
| # T41 | Natalia | Review Delivery Doc class | Badge Tick1 with solid fill |
| # T42 | Ronan | Review Driver class | Badge Tick1 with solid fill |
| # T43 | Natalia | Review Order class | Badge Tick1 with solid fill |

### Increment 3: Link / Integrate / Test Application

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Assignee | Description | Completed |
| Increment 3: WK 10-11 | | | |
| # T44 | Natalia | Work on CMD   1. Login 2. Newsagent CRUD 3. Admin CRUD | Badge Tick1 with solid fill |
| # T45 | Ivan | Work on CMD   1. Invoice CRUD 2. Order CRUD 3. Driver CRUD | Badge Tick1 with solid fill |
| # T46 | Ronan | Work on CMD   1. Publication CRUD 2. Delivery Docket CRUD | Badge Tick1 with solid fill |
| # T47 | Natalia/Ronan/Ivan | Test the whole application | Badge Tick1 with solid fill |

### Increment 4: Develop Instruction / Presentation

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Assignee | Description | Completed |
| Increment 4: WK 12-13 | | | |
| # T49 | Natalia | Present Fully Working Application | TBC |

# Known Defects/Bugs

**NOTE:**

There are a few known issues, defects, and potential improvements that require attention. The following list consists of identified concerns that we are aware of.

## Newsagent

1. ORDER update - error updating order, unknown column "title" in "field list"

2. PUBLICATION - no verification if publication exists before GET/UPDATE/DELETE

3. INVOICE - is not automatically generated/delivered to customer at the end of the month

4. CUSTOMER – active/inactive status not implemented

## Admin

1. No validation if user exists for update/delete

## Driver

1. UPDATE STOCK - no verification when stock < 0 during deduction

2. UPDATE STOCK - when book is OUT OF STOCK, it won’t let to add it back in

3. Upon login, app could display driver ID

## Unit Tests

1. INVOICE/DRIVER testing should be better