

**DESIGN AND TESTING 2**

**SPRINT 1**

### 

|  |
| --- |
| **Group 2** |
| Guerrero Cuenca, Claudia |
| Macarro Klepsch, Miguel |
| Volante González, José Manuel |

|  |
| --- |
| **Content** |
| First the level chosen for our project will be mentioned.  Then show a summary of the full description of this. Finally, this summary is specified with the different entities and features that will be implemented in our application. |

**Content**

1. Project description
   1. General description of the web application
   2. Entities
   3. User stories
   4. Features to modify in the existing system
   5. Specification with examples
2. Planning for the following sprints

**1.- PROJECT DECRIPTION**

**a.- General description of the web application**

Choice type and level:

Extension of the example provided, level 2 (min. 5 entities + 20 user stories).

Summary:

Our project consists in extending the project provided by the professors. The main features we are going to improve or implement are:

* The way visits are used: Instead of just having the owner or an administrator register the date of a visit, we will implement a fully automated scheduling system. An owner will be able to select a free timeslot with a veterinary of his choice, which will then be registered in the system.
* Payment information for a visit, including credit card data if applicable, will be stored in the system.
* In order to estimate the duration of a visit (for the scheduling) and its cost (for the payment system), we will categorize the visits by type (operation, revision, consultation)
* Veterinarians will be able to add a diagnosis to a visit, including prescriptions for medications.

**b.- Entities**

Entities to modify

**Visit**

Modified attributes:

* date: renamed to moment. Should represent a date with time

New relationships and attributes:

* Vet (many to one)
* VisitType (many to one)
* Diagnosis (one to one, optional)
* Paymnet (one to one, optional

New entities

These are the new entities that will be implemented with their attributes:

**VisitType** 🡪 Name, Price, Duration.

The default types are: consultation (20€, 30 min), revision (15€, 30 min) and operation (100€, 60 min).

**Diagnosis** 🡪 Moment, Description.

Relationships: Prescription (one to many)

**Prescription** 🡪 Frequency, Quantity.

Relationships: Medicine (many to one)

**Medicine** 🡪 Name, Business.

**Payment** 🡪 Date and time, Quantity), Method.

Relationships: CreditCard (one to one, optional), Secretary (many to one)

**CreditCard** 🡪 Holder, Brand, Number, ExpMonth, ExpYear, SecurityCode.

Relationships: None

**Secretary**

Extends from Person and has an associated user. This kind of user is in charge of registering payments.

The relationship between the classes can be seen in the following UML diagram. New classes that we will implement are highlighted in yellow.



**c.- User stories**

User stories that we will implement in our application:

Visit scheduling

(Visit, VisitType, Vet, Pet, Owner)

**User story #1**

**Feature**: Schedule an appointment online

**As a** pet owner

**So that** I don’t have to call

**I want** to be able to schedule appointments online.

(Involved entities: Owner, Pet, Visit)

**User story #2**

**Feature**: Request a visit with a specific veterinarian

**As a** pet owner

**So that** I can receive service from a veterinarian that had treated my pet before

**I want** to be able to request a visit with a specific veterinarian

(Involved entities: Owner, Pet, Vet)

**User story #3**

**Feature**: Manage appointments automatically

**As a** clinic owner

**So that** I can automate the scheduling process

**I want** the system to manage appointments for visits automatically, including making sure that no appointment is made outside of working hours (8:00 am – 8:00 pm, Monday through Friday) and that no vet has two visits scheduled at the same time

(Involved entities: Visit, Vet)

**User story #4**

**Feature**: Select a type of visit

**As a** clinic owner

**So that** time and visits are better utilized

**I want** that the owner of a pet can select a type of visit, which has an approximate duration.

(Involved entities: Visit, Owner, VisitType)

**User story #5A**

**Feature**: Upcoming visits view (pet owner)

**As a** pet owner

**So that** I won’t forget an appointment

**I want** a view that shows the visits I have scheduled in the future

(Involved entities: Visit, Owner)

**User story #5B**

**Feature**: Past visits view (pet owner)

**As a** pet owner

**So that** I can see the visits I made with my pets and the diagnosis for each one

**I want** a view that shows the visits I have had in the past and the diagnosis for each one

(Involved entities: Visit, Pet, Diagnosis, Prescription, Medicine, Owner)

**User story #6**

**Feature**: Upcoming visits view (vet)

**As a** vet

**So that** I know what I’ll have to do in a given week

**I want** a view that shows the visits I have scheduled by week

(Involved entities: Visit, Vet)

Payment registration

(Visit, Payment, CreditCard, Secretary)

**User story #7**

**Feature**: Register a payment with credit card or cash

**As a** pet owner

**So that** I can have the freedom of choosing how to pay for my visit

**I want** to be able to pay with credit card or cash

(Involved entities: Visit, Payment, CreditCard)

**User story #8**

**Feature**: Validate credit card

**As a** clinic owner

**So that** I can guarantee that all payments are registered correctly

**I want** that all credit cards introduced in the system are validated. No payment should be stored with an expired credit card or one that has an incorrect number

(Involved entities: Visit, Payment)

**User story #9**

**Feature**: Store who registered a payment

**As a** clinic owner

**So that** I can make sure no fraudulent payments are registered

**I want** that every payment includes the secretary that registered it

(Involved entities: Visit, Payment, CreditCard, Secretary)

**User story #10**

**Feature**: Suggest price for a visit based on its type

**As a** secretary

**So that** I know how much to charge a client

**I want** that the price of a visit is suggested based on its type.

(Involved entities: Visit, VisitType)

**User story #11**

**Feature**: Freely assign price to a visit

**As a** clinic owner

**So that** the secretaries can adjust the price of a visit to specific circumstances

**I want** that when registering a payment, the price is already filled in based on the visit type, but can be changed manually

(Involved entities: Visit, Payment, VisitType)

**User story #12**

**Feature**: View all unpaid visits

**As a** secretary

**So that** I know which visits have not been paid yet

**I want** a view that lists all unpaid visits ordered by moment and which includes links to each visit so that I can pay them

(Involved entities: Visit, Payment)

Diagnosis registration

(Visit, Diagnosis, Prescription, Medicine)

**User story #13**

**Feature**: Add diagnosis to a visit

**As a** vet

**So that** I can later consult the medical history of a pet

**I want** to be able to add a diagnosis to a visit

(Involved entities: Visit, Diagnosis)

**User story #14**

**Feature**: Add prescriptions to a diagnosis

**As a** vet

**So that** I can tell the owner what medicine to give their pets

**I want** to be able to add prescriptions to a diagnosis.

(Involved entities: Diagnosis, Prescription, Medicine)

**User story #15**

**Feature**: Select medicine from database

**As a** vet

**So that** I can make sure I don’t prescribe a medicine that doesn’t exist or is spelled differently

**I want** to select a medicine from a list of medicines stored in the system when registering a prescription

(Involved entities: Prescription, Medicine)

Medicine registration

(Medicine)

**User story #16**

**Feature**: Add new medicine the system

**As an** administrator

**So that** vets can prescribe new medicine

**I want** add new medicine to the system

(Involved entities: Medicine)

**User story #17**

**Feature**: Edit or delete the medicines in the system

**As an** administrator

**So that** I can correct potential mistakes

**I want** to be able to edit or delete medicines (as long as they haven’t been prescribed yet)

(Involved entities: Medicine, Prescription)

Admin dashboard

(Visit, Payment, Diagnosis)

**User story #18**

**Feature**: View revenue by month

**As an** administrator

**So that** I can know how the clinic is doing economically

**I want** a view that shows the total revenue (sum of all payments) by month

(Involved entities: Visit, Payment)

**User story #19**

**Feature**: See all the characteristics of visits already made

**As an** administrator

**So that** I can check the correct functioning of the clinic

**I want** a view with all the visits already made that includes the features of that visits and a link for their diagnosis and payment.

(Involved entities: Visit, Payment, Diagnosis)

VisitType management

(VisitType)

**User story #20**

**Feature**: Add new types of visit

**As a** admin

**So that** the system can be customized to the necessities of the clinic

**I want** to be able to add new types of visit

(Involved entities: VisitType)

**User story #21**

**Feature**: Edit types of visit

**As a** admin

**So that** the prices of the services that are offered can be changed but the scheduling doesn’t become inconsistent

**I want** to be able to edit only the price of a type of visit, and that the duration can not be edited once the type is created

(Involved entities: VisitType)

**d.- Features to modify in the existing system**

* When login as Vet, in Owner tab, an exception is raised.
* There are many owners for the same user. Limit to 1. In the list of owners put his personal data.
* A user not authenticated can’t do anything.
* Change logo and add something on the home page.
* Change association from user->owner to user->person

**e.- Specification with examples**

User story #1 (Schedule an appointment online)

**[US1-P1] Schedule correct appointment (Positive scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a vet named* Antonio Sánchez *that has a free slot* 2020-08-03 11:00 am*; a visit type named* consultation.

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, as vet ‘Antonio Sánchez’, and as type ‘consultation’
4. There should be a view that shows a weekly calendar in form of a table with all free slots for the given vet
5. Select a free slot, 2020-08-03 11:00 am

Result: The visit is stored in the system. It should be visible to the owner in the upcoming visits view. If another or the same owner want to schedule a visit, the slot should be taken.

**[US1-N1] Try to schedule 2 visits at the same time for the same vet (Negative scenario)**

*Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; another user that is an owner, with name* owner2 *and password* owner2*, that owns a pet named* Mario; *a vet named* Antonio Sánchez *that has a free slot* 2020-08-03 12:00 am*; a visit type named* consultation*; another visit type named* revision.

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, as vet ‘Antonio Sánchez’, and as type ‘consultation’
4. In a different window, log in as owner2 (password: owner2)
5. Go to the view for scheduling a new visit
6. Select as pet ‘Mario, as vet ‘Antonio Sánchez’, and as type ‘revision’
7. Select the slot 2020-08-04 12:00 am and confirm
8. As owner1, select the same slot (2020-08-04 12:00 am) and confirm

Result: A error message should be shown.

**[US1-N2] Owner without pet tries to make an appointment (Negative scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner3 *and password* owner3*, that doesn’t have any pets.*

1. Log into the platform as owner3 (password: owner3)
2. Go to the view for scheduling a new visit

Result: A message should be shown telling the owner that he has to add a pet first

**Summary of scenarios for US1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | **Owner has pet?** | **Slot occupied?** | **Result** |
| US1-P1 | Yes | No | Success |
| US1-N1 | Yes | Yes | Error |
| US1-N2 | No | N/A | Error |

User story #2 (Request a visit with a specific veterinarian)

**[US2-P1] Schedule correct appointment (Positive scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a vet named* Antonio Sánchez *that has a free slot* 2020-08-03 1:00 pm*; a visit type named* consultation.

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, as vet ‘Antonio Sánchez’, and as type ‘consultation’
4. There should be a view that shows a weekly calendar in form of a table with all free slots for the given vet
5. Select a free slot, 2020-08-03 1:00 pm

Result: The visit is stored in the system. It should be visible to the owner in the upcoming visits view. If another or the same owner want to schedule a visit, the slot should be taken.

**[US2-N1] Send a POST request with a vet that doesn’t exist (Negative scenario)**

*Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a visit type named* consultation. *No vet named* Invalid vet *should exist in the system.*

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, and as type ‘consultation’
4. Select as vet ‘Invalid vet’, a vet that doesn’t exist tin the dropdown menu, for example by editing the html form manually
5. There should be a view that shows a weekly calendar in form of a table with all free slots for the given vet
6. Select a free slot, 2020-08-03 12:00 pm

Result: An exception should be shown.

**Summary of scenarios for US2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | **Vet selected** | **Vet exists?** | **Result** |
| US2-P1 | Antonio Sánchez | Yes | Success |
| US2-N1 | Invalid vet | No | Error |

User story #3 (Manage appointments automatically)

**[US3-P1] Schedule a correct appointment (Positive scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a vet named* Antonio Sánchez *that has a free slot* 2020-08-03 2:00 pm*; a visit type named* consultation.

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, as vet ‘Antonio Sánchez’, and as type ‘consultation’
4. There should be a view that shows a weekly calendar in form of a table with all free slots for the given vet
5. Select a free slot, 2020-08-03 2:00 pm

Result: The visit is stored in the system. It should be visible to the owner in the upcoming visits view. If another or the same owner want to schedule a visit, the slot should be taken.

**[US3-N1] Try to schedule an appointment outside of working hours (Negative scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a vet named* Antonio*; a visit type named* consultation.

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, as vet ‘Antonio Sánchez’, and as type ‘consultation’
4. There should be a view that shows a weekly calendar in form of a table with all free slots for the given vet
5. Select a slot outside of working hours, 2020-08-03 02:00 am, for example by editing the html form manually

Result: An exception should be shown

**Summary of scenarios for US3**

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | **Slot** | **Slot outside of working hours?** | **Result** |
| US3-P1 | 2020-08-03 2:00 pm | No | Success |
| US3-N1 | 2020-08-03 2:00 am | Yes | Error |

User story #4 (Select a type of visit)

**[US4-P1] Schedule a correct appointment (Positive scenario)**

*Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a vet named* Antonio Sánchez *that has a free slot* 2020-08-03 3:00 pm*; a visit type named* consultation*.*

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, as vet ‘Antonio Sánchez’, and as type ‘consultation’
4. There should be a view that shows a weekly calendar in form of a table with all free slots for the given vet
5. Select a free slot, 2020-08-03 3:00 pm

Result: The visit is stored in the system. It should be visible to the owner in the upcoming visits view. If another or the same owner want to schedule a visit, the slot should be taken.

**[US4-N1] Send a POST request with a nonexistent visit type (Negative scenario)**

*Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a vet named* Antonio Sánchez *that has a free slot* 2020-08-03 4:00 pm*. No visit type names* Invalid type *should exist in the system.*

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view for scheduling a new visit
3. Select as pet ‘Pepe’, and as vet ‘Antonio Sánchez’
4. Select as type ‘Invalid type’, a type that doesn’t exist in the dropdown menu, for example by editing the html form
5. There should be a view that shows a weekly calendar in form of a table with all free slots for the given vet
6. Select a free slot, 2020-08-03 4:00 pm

Result: An exception should be shown.

**Summary of scenarios for US4**

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | **Visit type** | **Visit type exists?** | **Result** |
| US4-P1 | consultation | Yes | Success |
| US4-N1 | Invalid type | No | Error |

User story #5A (Upcoming visits view (pet owner))

**[US5A-P1] At least one visit (Positive scenario)**

*Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a visit type named* consultation*; a vet named* Antonio Sánchez *that has a consultation scheduled with Pepe for* 2020-09-01 10:00 am*.*

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view that shows the upcoming visits

Result: A visit for 2020-09-01 10:00 am should be shown.

**[US5A-N1] No visits (Negative scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner2 *and password* owner2*, that doesn’t have any visits scheduled in the future.*

1. Log into the platform as owner2 (password: owner2)
2. Go to the view that shows the upcoming visits

Result: A message should be shown telling the user that he has no future visits scheduled.

**[US5A-N2] Checking that a past visit doesn’t show up in the future visits view (Negative scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a visit type named* consultation*; a vet named* Antonio Sánchez *that has a consultation scheduled with Pepe for* 2020-02-03 10:00 am*.*

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view that shows the upcoming visits

Result: The visit for 2020-02-03 10:00 should not be shown since it is in the past.

**Summary of scenarios for US5A**

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Visit** | **Result** |
| US5A-P1 | Visit in the future | The visit it shown |
| US5A-N1 | No visits | No visit is shown |
| US5A-N2 | Visit in the past | The visit is not shown (but other visits that are in the future might be shown |

User story #5B (Past visits view (pet owner))

**[US5B-P1] At least one visit (Positive scenario)**

*Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a visit type named* consultation*; a vet named* Antonio Sánchez *that has a consultation scheduled with Pepe for* 2020-02-03 10:00 am*.*

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view that shows the past visits

Result: A visit for 2020-02-03 10:00 am should be shown.

**[US5B-N1] No visits (Positive scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner2 *and password* owner2*, that doesn’t have any visits scheduled in the past.*

1. Log into the platform as owner2 (password: owner2)
2. Go to the view that shows the past visits

Result: A message should be shown telling the user that he has not made any visits in the past.

**[US5B-N2] Checking that a future visit doesn’t show up in the past visits view (Negative scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a visit type named* consultation*; a vet named* Antonio Sánchez *that has a consultation scheduled with Pepe for* 2020-09-01 10:00 am*.*

1. Log into the platform as owner1 (password: 0wn3r)
2. Go to the view that shows the past visits

Result: The visit for 2020-09-01 10:00 should not be shown since it is in the future.

**Summary of scenarios for US5B**

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Visit** | **Result** |
| US5B-P1 | Visit in the past | The visit it shown |
| US5B-N1 | No visits | No visit is shown |
| US5B-N2 | Visit in the future | The visit is not shown (but other visits that are in the past might be shown |

User story #6 (Upcoming visits view (vet))

**[US6-P1] One visit (Positive scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1*, that owns a pet named* Pepe*; a visit type named* consultation*; a vet named* Antonio Sánchez, *with username* vet2 *and password* vet2, *that has a consultation scheduled with Pepe for* 2020-09-01 10:00 am*.*

1. Log in as vet2 (password: vet2)
2. Go to the view that shows the upcoming visits for vets

Result: The consultation with Pepe on 2020-09-01 10:00 should be shown.

**[US6-N1] No visits (Negative scenario)***Context: It is assumed that the system already has the following data stored: A vet* *with username* vet3 *and password* vet3, *that doesn’t have any visits scheduled in the future.*

1. Log in as vet3 (password: vet3)
2. Go to the view that shows the upcoming visits for vets

Result: A message should be shown telling the vet that he has no future visits scheduled.

**[US6-N2] Checking that a past visit doesn’t show up in the future visits view (Negative scenario)***Context: It is assumed that the system already has the following data stored: A user that is an owner, with name* owner1 *and password* 0wn3r*, that owns a pet named* Pepe*; a visit type named* consultation*; a vet named* Antonio Sánchez *that has a consultation scheduled with Pepe for* 2020-09-01 10:00 am*.*

1. Log into the platform as vet2 (password: vet2)
2. Go to the view that shows the past visits

Result: The visit for 2020-09-01 10:00 should not be shown since it is in the future.

**Summary of scenarios for US6**

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Visit** | **Result** |
| US6-P1 | Visit in the future | The visit it shown |
| US6-N1 | No visits | No visit is shown |
| US6-N2 | Visit in the past | The visit is not shown (but other visits that are in the future might be shown |

User story #7 (Register a payment with credit card or cash)

**US7-P1 (Positive scenario) – Pay cash**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. Fill the fields of a payment: ‘method’=cash, ‘moment’=2020-02-02 10:30, ‘quantity’=50
6. Press button to accept the payment

Result: The data are saves in the database like the payment of that visit. It’s pay by cash, so no problem in the system and the owner has to pay the quantity by hand.

**US7-P2 (Positive scenario) – Pay credit card**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. Fill the fields of a payment: ‘method’=creditcard, ‘moment’=22/02/20 10:30, ‘quantity’=50
6. Press the button “Add credit card” to add a credit card for this payment
7. Fill the fields of a valid credit card: ‘holder’=Jean Coleman, ‘brand’=visa, ‘number’= 4387829024013796, ‘expMonth’=05, ‘expYear’=22, ‘securityCode’=255
8. Press button to accept credit card
9. Press button to accept payment

Result: The data are saves in the database like the payment of that visit. It’s pay by credit card so the system saves the data of this credit card too. Data is saving correctly.

**US7-N1 (Negative scenario)**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. Fill the fields of a payment: ‘method’=creditcard, ‘moment’=22/02/20 10:30, ‘quantity’=50
6. Doesn´t add a credit card
7. Press button to accept the payment

Result: The data of this payment hasn’t been saved. Secretary can see a message that say “Need a credit card to do a payment with this method”.

User story #8 (Validate credit card)

**US8-P1 (Positive scenario)**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. Fill the fields of a payment: ‘method’=creditcard, ‘moment’=22/02/20 10:30, ‘quantity’=50
6. Press the button “Add credit card” to add a credit card for this payment
7. Fill the fields of a valid credit card: ‘holder’=Jean Coleman, ‘brand’=visa, ‘number’= 4387829024013796, ‘expMonth’=05, ‘expYear’=22, ‘securityCode’=255
8. Press button to accept the credit card

Result: The credit card number is correctly and the expiration too. The data is saved in the database correctly.

**US8-N1 (Negative scenario)**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. Fill the fields of a payment: ‘method’=creditcard, ‘moment’=22/02/20 10:30, ‘quantity’=50
6. Press the button “Add credit card” to add a credit card for this payment
7. Fill the fields of a valid credit card: ‘holder’=Jean Coleman, ‘brand’=visa, ‘number’= 2200554488996655, ‘expMonth’=05, ‘expYear’=22, ‘securityCode’=255
8. Press button to accept the credit card

Result: The credit card number is not correctly so the system validate this number and shows a message that say “This credit card number is not valid”.

**US8-N2 (Negative scenario)**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. Fill the fields of a payment: ‘method’=creditcard, ‘moment’=22/02/20 10:30, ‘quantity’=50
6. Press the button “Add credit card” to add a credit card for this payment
7. Fill the fields of a valid credit card: ‘holder’=Jean Coleman, ‘brand’=visa, ‘number’= 4387829024013796, ‘expMonth’=05, ‘expYear’=15, ‘securityCode’=255
8. Press button to accept the credit card

Result: The credit card expiration is not correctly so the system validate this number and shows a message that say “This credit card expiration passed”.

User story #9 (Store who registered a payment)

**US9-P1 (Positive scenario)**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. Fill the fields of a payment: ‘method’=creditcard, ‘moment’=22/02/20 10:30, ‘quantity’=50
6. Press the button “Add credit card” to add a credit card for this payment
7. Fill the fields of a valid credit card: ‘holder’=Jean Coleman, ‘brand’=visa, ‘number’= 4387829024013796, ‘expMonth’=05/22
8. Press button to accept credit card
9. Press button to accept payment

Result: This payment is made by the user “secretary1”. The system knows that this user has a role of Secretary of the system. So this person can do the payment correctly.

**US9-N1 (Negative scenario)**

1. Log into the platform as owner1 (password: 0wn3r)
2. Attend to access the URL of the unpaid visits view for secretary without being logged in as an owner

Result: This payment was going to be performed by the user “owner1”. The system knows that this user has a role of Owner of the system. So an error is appeared.

**US9-N2 (Negative scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Attend to access the URL of the unpaid visits view for secretary without being logged in as an owner

Result: This payment was going to be performed by the user “vet2”. The system knows that this user has a role of Vet of the system. So an error is appeared.

User story #10 (Suggest price for a visit based on its type)

**US10-P1 (Positive scenario)**

*Assuming that the VisitType that the owner requested is consultation (20€).*

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. The fields of a payment are blank, exception of ‘quantity’=20 because 20€ is the estimate price for this visit type

Result: Secretary can see the estimate price in the field of quantity. It’s the price that has that visit type.

**US10-N1 (Negative scenario)**

*Assuming that the VisitType that the owner requested is* free *(0€).*

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. The fields of a payment are blank, exception of ‘quantity’=0.00 because 0€ is the estimate price for this visit type

Result: Secretary can see the estimate price in the field of quantity. It’s the price that has that visit type.

User story #11 (Freely assign price to a visit)

**US11-P1 (Positive scenario)**

*Assuming that the VisitType that the owner requested is consultation (20€).*

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. The fields of a payment are blank, exception of ‘quantity’=20 because 20€ is the estimate price for this visit type
6. Secretary change this field ‘quantity’=70
7. Secretary fills the others fields: ‘method’=cash, ‘moment’=22/02/20 10:30
8. Press button to accept the payment

Result: Secretary can change the quantity. The price of a visit type is only approximate but then secretary calculates the real price. This data saves in the database correctly.

**US11-N1 (Negative scenario)**

*Assuming that the VisitType that the owner requested is consultation (20€).*

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. Select the first one that is the next visit that secretary have to manage
4. Press button of “Do payment”
5. The fields of a payment are blank, exception of ‘quantity’=20 because 20€ is the estimate price for this visit type
6. Secretary change this field ‘quantity’=setentaeuros
7. Secretary fills the others fields: ‘method’=cash, ‘moment’=22/02/20 10:30
8. Press button to accept the payment

Result: An error is appeared in the form of the payment, because secretary adds a quantity that is a string. Quantity has to do a integer.

User story #12 (View all unpaid visits)

**US12-P1 (Positive scenario)**

1. Log into the platform as secretary1 (password: s3c3tary)
2. Go to the view with the list of all visits without a payment
3. If you select one of that can see the button to pay it

Result: In this view a secretary can see all visits without a payment. Can select one and do her payment. Only a user with the Secretary role can see this view.

**US12-N1 (Negative scenario)**

1. Log into the platform as owner1 (password: 0wn3r)
2. Attend to access the URL of the unpaid visits view for secretary without being logged in as an owner

Result: Only a user with the Secretary role can see this view. So an owner can’t see this view with the list of all visits without a payment.

**US12-N2 (Negative scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Attend to access the URL of the unpaid visits view for secretary without being logged in as a vet

Result: Only a user with the Secretary role can see this view. So a vet can’t see this view with the list of all visits without a payment.

User story #13 (Add diagnosis to a visit)

**US13-P1 (Positive scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Go to the view for adding a diagnosis to a visit
3. Introduce as moment ‘2020-08-03 2:00 pm’ and as description ‘Sample description’ and confirm the addition

Result: The newly added diagnosis should be shown on the visit

**US13-N1 (Negative scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Go to the view for adding a diagnosis to a visit
3. Introduce as moment ‘2020-08-03 2:00 pm’ and leave the description empty
4. Attempt to confirm the addition

Result: It should not be possible to add the diagnosis to the visit.

**US13-N2 (Negative scenario)**

*Assuming that the visit is already diagnosed.*

1. Log into the platform as vet2 (password: vet2)
2. Go to the view for adding a diagnosis to a visit
3. Introduce as moment ‘2020-08-03 2:00 pm’ and as description ‘Sample description’ and confirm the addition
4. Attempt to confirm the addition

Result: It should not be possible to add the diagnosis to the visit.

User story #14 (Add prescription to a diagnosis)

**US14-P1 (Positive scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Go to the view for adding a diagnosis to a visit
3. Introduce in the section ‘Prescription’ as frequency ‘Twice a day’, as duration ‘Two weeks’, as medicine ‘Betadine’ and confirm the addition

Result: The newly added prescription and medicine should be shown on the diagnosis

**US14-N1 (Negative scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Go to the view for adding a diagnosis to a visit
3. Introduce in the section 'Prescription' as frequency ‘Twice a day’, leave the duration empty, as medicine ‘Betadine’ and confirm the addition
4. Attempt to confirm the addition

Result: It should not be possible to add the prescription to the diagnosis

User story #15 (Select medicine from database)

**US15-P1 (Positive scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Go to the view for adding a diagnosis to a visit
3. Introduce in the section ‘Prescription’ as frequency ‘Twice a day’, as duration ‘Two weeks’, as medicine select ‘Betadine’ and confirm the addition

Result: The newly added prescription and medicine should be shown on the diagnosis

**US15-N1 (Negative scenario)**

1. Log into the platform as vet2 (password: vet2)
2. Go to the view for adding a diagnosis to a visit
3. Introduce in the section ‘Prescription’ as frequency ‘Twice a day’, as duration ‘Two weeks’, as medicine select ‘Hacking’, for example by editing the html manually and confirm

Result: An exception should be shown

User story #16 (Add new medicine the system)

**US16-P1 (Positive scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view for adding a new medicine
3. Introduce as name ‘Sample drug’ and as brand ‘Sample brand’ and confirm the creation

Result: The newly created medicine should show up in the list of all medicines

**US16-N1 (Negative scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view for adding a new medicine
3. Introduce as name ‘Sample drug’ and leave the brand empty
4. Attempt to confirm the creation

Result: It should not be possible to create the new medicine

User story #17 (Edit or delete the medicines in the system)

**US17-P1 (Positive scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view for adding a new medicine
3. Introduce as name ‘Unused drug’ and as brand ‘Sample brand’ and confirm the creation
4. Go to the view that lists medicine
5. Select the medicine named ‘Unused drug’
6. Delete the medicine named ‘Unused drug’

Result: The medicine should be correctly deleted from the system

**US17-N1 (Negative scenario)***For this scenario it is assumed that a medicine with name* Betadine *exists in the system and that it has been prescribed at least once.*

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view that lists medicine
3. Select the medicine named ‘Betadine’
4. Attempt to delete the medicine named ‘Betadine’

Result: An error should be thrown as this medicine has already been prescribed.

User story #18 (View revenue by month)

**US18-P1 (Positive scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view in dashboard for seeing the revenue by month

Result: The view that shows the total revenue (sum of all payments) by month is displayed.

**US18-N1 (Negative scenario)**

1. Attempt to access the URL of the view revenue by month without being logged in as an admin

Result: An error message should be shown

User story #19 (See all the characteristics of visits already made)

**[US19-P1] Show list of all visits (Positive scenario)***Context: It is assumed that the system already has the following data stored: An admin with name* admin1 *and password* 4dm1n.

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view in dashboard for seeing all the visits already made

Result: The view that shows all the visits already made that includes the features of that visits and a link for their diagnosis and payment is displayed

**[US19-N1] Attempt to show the list of all visits without authorization (Negative scenario)**

1. Attempt to access the URL of the view in dashboard for seeing all the visits already made without being logged in as an admin

Result: An error message should be shown

**Summary of scenarios for US19**

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | **User** | **Is the user authorized?** | **Result** |
| US19-P1 | admin1 | Yes | The view it shown |
| US19-N1 | None (unauthenticated) | No | Error message |

User story #20 (Add new types of visit)

**US20-P1 (Positive scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view for adding a new type of visit
3. Introduce as name ‘Sample visit type’, as duration ‘1’, as price ‘1’ and confirm the creation

Result: The newly created visit type should show up in the list of all visit types

**US20-N1 (Negative scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view for adding a new type of visit
3. Introduce as name ‘Sample visit type’, as duration ‘1’ and leave the price empty
4. Attempt to confirm the creation

Result: It should not be possible to create the new visit type

User story #21 (Edit types of visit)

**US21- P1 (Positive scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view for adding a new type of visit
3. Introduce as name ‘Unused visit type’, as duration ‘1’, as price ‘1’ and confirm the creation
4. Go to the view that lists visit type
5. Select and edit the visit type named ‘Unused visit type’
6. Introduce as price ‘2’ and confirm the edition

Result: The visit type should be correctly edited

**US21-N1 (Negative scenario)**

1. Log into the platform as admin1 (password: 4dm1n)
2. Go to the view for adding a new type of visit
3. Introduce as name ‘Unused visit type’, as duration ‘1’, as price ‘1’ and confirm the creation
4. Go to the view that lists visit type
5. Select and edit the visit type named ‘Unused visit type’
6. Introduce as duration ‘2’ and confirm the edition

Result: It should not be possible to edit the visit type

**2.- PLANNING FOR THE FOLLOWING SPRINTS**

|  |  |  |
| --- | --- | --- |
| **SEMANA** | **PROPIETARIO** | **TRABAJO** |
| 2 marzo – 8 marzo | Miguel | Hacer correcciones proyecto base (Implementar cambio user 🡪 Person, Cambiar muchos Owner por 1), Implementar US1, US2 y US4 |
| Claudia | Implementar US7, US8 y US9 |
| Josema | Implementar US13, US14 y US15 |
| 9 marzo – 15 marzo | Miguel | Implementar US3, US5A, US5B |
| Claudia | Implementar US10, US11 y US12 |
| Josema | Implementar US16 y US17 |
| 16 marzo – 22 marzo | Miguel | Pruebas unitarias para correcciones, US1 – US5B |
| Claudia | Pruebas unitarias para US7 – US12 |
| Josema | Pruebas unitarias para US13 – US17 |
| 23 marzo – 29 marzo | Miguel | Automatización de pruebas con Travis |
| Claudia |
| Josema |
| **ENTREGA SPRINT 2** | | |
| 30 marzo – 5 abril | Miguel | Implementar US6 |
| Claudia | Implementar US18 |
| Josema | Implementar US20 y US21 |
| 6 abril –  12 abril | Miguel | Pruebas unitarias para US6 |
| Claudia | Pruebas unitarias para US18 |
| Josema | Pruebas unitarias para US20 y US21 |
| 13 abril – 19 abril | Miguel | Pruebas de integración |
| Claudia |
| Josema |
| 20 abril – 26 abril | Miguel | Pruebas de integración |
| Claudia |
| Josema |
| 27 abril –  3 mayo | Miguel | Pruebas de interfaz de usuario  (de lo implementado hasta el momento) |
| Claudia |
| Josema |
| 4 mayo – 10 mayo | Miguel | Pruebas de interfaz de usuario  (de lo implementado hasta el momento) |
| Claudia |
| Josema |
| **ENTREGA SPRINT 3** | | |
| 11 mayo – 17 mayo | Miguel | Implementar US19  Pruebas unitarias para US19 |
| Claudia | Corregir error Vet 🡪 Owner  Pruebas unitarias para ello |
| Josema | Inicio sin loguear  Estética de la página  Pruebas unitarias para ello |
| 18 mayo – 24 mayo | Miguel | Pruebas de integración (si procede)  Pruebas de interfaz de usuario  (las restantes) |
| Claudia |
| Josema |
| 25 mayo – 31 mayo | Miguel | Pruebas end-to-end en los contoladores |
| Claudia |
| Josema |
| 1 junio –  5 junio | Miguel | Pruebas de rendimiento y refactorizaciones |
| Claudia |
| Josema |
| **ENTREGA SPRINT 4** | | |