Issey

User stories: 5, 7, 8, 10,11, 13, 17, 16, 19, 21, 23

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 5 | A Teacher | A log in portal / client access page | So that I can manage my students and lessons. Update my personal details.  Advise if I am available for a lesson or not (booked out, sick, etc).   1. The user (Teacher) can register via a form by implementing the Django Authentication System and the Forms API 2. The user will have their own unique username and password created by implementing the Django User and Forms API 3. The user will have access to information set consequently via the User API i.e. the teacher has staff level permission to view records of other Teachers and Students 4. user can log in via the Admin interface with admin (staff) permissions provided via the Django User API (address/admin) 5. user can query their own records (personal information) and their Students records via the Admin interface by implementing the Model API   **Acceptance Criteria**   * User understands how to navigate the Admin interface via training * Communication with the database backend for user * Can query records from the database backend * Test driven development practices, unit tests pass | Must | 9 |
| 7 | A Teacher | An application page or apply to become a teacher | I can apply to become a music teacher and complete any specific processes required in accordance with the Music School   1. An Application page implementing the Django Authentication System, Forms API and template 2. Configured the URL mappings via the Url API (url pattern) necessary web responses via the HTTP API 3. The form contains text fields for contact information e.g. Name, Address, Phone, Email 4. The form contains select fields for a list or input fields as check boxes for a given specialization e.g. Trombone, Violin, Guitar, Voice 5. The form contains select fields for a list or input fields as check boxes for skill level e.g. intermediate, advanced 6. The form contains a text field for number of years teaching 7. All fields have sanity/validation checking via the Forms API 8. Form information is parsed and emailed to owner to be reviewed and/or persisted (added) as a record to the database (Enquiry relation)   **Acceptance Criteria**   * Application page is available and accessible * Application page contains the teacher application form * Sanity check, validation, Exception handling * Email is sent to owner and/or form information is persisted (processed) * Navigation * Test driven development, all unit tests pass | Must | 8 |
| 8 | A Student or Parent | An “About Us” page | I can find out what my child’s/children’s lessons will consist of, what instruments my child/children could learn, how long the lessons can run for, what hours of the day can lessons be booked, what days of the week can lessons be booked, how qualified teachers are, details on each teacher hired at Pinelands Music School, what learning outcomes will my child/children receive after taking the lessons, policies, and if I can accompany my child/children in his/her/their lesson.   1. The About Us contains an article element 2. Infromation relating to the music school can be returned from the database by implementing the Model API and template 3. Alternatively information can be static in a simple article element and table elements displaying relevant information 4. Background information for the music school in the article element 5. Opening hours in a table element 6. Information regarding lessons in a table element 7. Profile card for each teacher within elements and their qualification information within div elements   **Acceptance Criteria**   * About Us page is available and accessible * About Us page displays information * Navigation * Sanity check, validation, Exception handling, web responses * Test driven development, all unit tests pass | Must | 6 |
| 10 | A Student or Parent | instrument hire page | I can hire an instrument required for my lesson appropriate to my needs and finances/budget.   1. Information is returned from the database and the Instrument relation using the Django Model API 2. Configured the URL mappings via the Url API (url pattern) and necessary web responses via the HTTP API 3. Information is displayed for availability of instrument in a using the Model API and template 4. Information regarding Instrument price is displayed 5. Information regarding Instrument availability is displayed 6. Information regarding Instrument hiring date(s) is displayed 7. A Form is available to hire a given instrument by implementing the Forms API and template 8. The form contains a text field for student identification 9. The form contains an input field as a radio buttons to hire an instrument; sanity check as only one instrument at a given time can be hired (radio button to only allow one item selected) 10. The form does not allow (will not post) if an instrument is unavailable   **Acceptance Criteria**   * Instrument page is available and accessible * Instrument page displays information from the Instrument relation * Instrument page contains hiring information from the Instrument relation * Sanity check, validation, Exception handling, web responses * Navigation * Form information is persisted (processed) or form does not post if unavailable * Test driven development, all unit tests pass | Must | 5 |
| 11 | The Owner of the business | Social media linkages | I can promote, advertise and market my business through the means of Facebook, Instagram, Linked In, etc.   1. The home page displays links placed in footer or within div elements 2. Information can be static in a simple article or div with links to Social media 3. Implement the Facebook API to display display friends of the music school from Facebook 4. Implement the LinkedIn API to display connections to the business   **Acceptance Criteria**   * Home page displays links to Social media * Home page displays Facebook friends * Home page displays LinkedIn connections | Could | 4 |
| 13 | A Teacher | A contact page | I can get in touch with the Music School via a communication method that is best suitable to me.   1. The user (teacher) can contact the owner/other staff via the provided Django Admin 2. Implement messaging/direct email via the Admin interface by implementing the Model API 3. Alternatively user can also contact owner/other staff via a form implementing the Forms API and template 4. A form for enquiry contains text fields for Teacher ID, Email, Mobile 5. Form contains a text area for called Message 6. All fields have sanity/validation checking via the Forms API 7. Form information is parsed and emailed to owner/other staff to be reviewed and/or persisted (added) as a record to the database (Enquiry relation)   **Acceptance Criteria**   * Contact page contains the contact form or user understands how to send a messages using the Admin interface via training * Sanity check, validation, Exception handling, web responses * Navigation * Email is sent to owner and/or form information is persisted (processed) * Test driven development, all unit tests pass | Could | 3 |
| 16 | The Owner of the business | Security for the website | The website can be protected from hackers and personal information will be kept safe.   1. Implement best practice for the Django framework using test driven development 2. User name validation provided via Django User or Forms API 3. Password validation and hashing provided via Django User or Forms API 4. Exception handling via the Django Core Exceptions API 5. Consider local storage solution vs cloud solution (AWS) for security; which option is most viable regarding sensitive client information 6. Maintenance and update of code to latest versions to avoid critical exploits 7. Avoiding legacy code/any technology to avoid critical exploits 8. Awareness of found critical exploits new and old e.g. maintenance cycle developers are aware of exploits and can attempt to remedy them 9. Updates/patches and downtime to remedy found exploits   **Acceptance Criteria**   * Best practices are followed regarding the Django Framework’s validation implementation for the Users and Forms API * Test driven development, all unit tests pass for validation and Exception handling * A maintenance cycle is implemented, software and code maintenance is part of the cycle * Users are aware of downtime to remedy exploits via training | Would | 4 |
| 17 | The Owner of the business | Backup storage procedure | If the database corrupts or database turns off by itself. The database will have a backup, so no data will be lost.   1. On site automated backup (database dump) to a local server performed daily or register backup and recovery plan via a cloud storage solution e.g. AWS storage solution 2. Recovery protocol is implemented if the system is compromised to inform users of possible downtime for recovery 3. Consider local storage solution vs cloud solution (AWS) for security; which option is most viable regarding sensitive client information 4. Consider local backup options vs the cloud storage solution regarding cost benefit   **Acceptance Criteria**   * Implemented either a local storage solution or cloud solution * A maintenance cycle is implemented, backup and storage is part of the cycle * Users are aware of recovery protocol and downtime via training | Would | 3 |
| 19 | A Student or Parent | A payment portal | I can pay my tuition fee using options such as PayPal, keying in credit/debit card details.   1. A payment webpage implementing the django-paypal API for a Paypal payment gateway 2. Configure the django-paypal API for persisted information to database and the Payment relation 3. Configured the URL mappings via the Url API (url pattern) and necessary web responses via the HTTP API 4. Set attributes e.g. item, invoice, currency type, amount etc. 5. A button will submit a form via post to Paypal for processing payment 6. Configured URL mappings to return to webpage via post after payment is processed or to return to the webpage if payment is cancelled or unsuccessful 7. Add template to process payment and web pages for complete, process, cancelled 8. Test PayPal Business account via the Sandbox is processing test payments 9. Test order is information is persisted to database   **Acceptance Criteria**   * Payment page is available and accessible * Payment is processed or cancelled * Returned to website regardless of a successful payment * Sanity check, validation, Exception handling, web responses * Navigation * Test driven development, all unit tests pass | Would | 4 |
| 21 | A Student or Parent | A “Reviews” tab | I can read reviews about Pinelands Music School before confirming my enrolment at Pinelands Music School.   1. A Reviews page to display testimonials from Students or Teachers placed in footer or either div elements in the home page 2. Information can be static an simple article element or div element displaying reviews with consent from Students or Teachers   **Acceptance Criteria**   * Reviews page or elements in home page display testimonial information * Sanity check, validation, Exception handling, web responses * Navigation * Test driven development, all unit tests pass | Would | 2 |
| 23 | The Owner of the business | Mobile cross-compatibility | I can provide a robust, cross platform/browser solution for clients and customers.   1. Test current functionality in other browsers e.g. Firefox, Safari, IE 2. Test current functionality for version support e.g. <https://caniuse.com/> 3. Consider issues with browser support e.g. backwards (legacy) compatibility 4. Re-implement code to support other browsers 5. Consider layout issues when scaling to new platforms and the “mobile first” practice is not implemented and implications to functionality i.e. CSS is not a definitive way to layout the website regardless of responsive web design 6. Consider how the current layout on the website will be displayed when scaling CSS, text, images 7. Consider an appropriate Framework/technology for mobile development e.g. Xamarin provides close to native support for all platforms both Android and IOS vs displaying information via a device’s browser 8. Layout design for mobile devices 9. Implement test driven development for the website on mobile devices   **Acceptance Criteria**   * Website functions correctly on alternative browsers * Website functions correctly on mobile devices * Sanity check, validation, Exception handling, web responses * Test driven development, all unit tests pass | Would | 6 |
|  |  |  |  | Total: | 120 |
| 2 | The Owner of the business | A database for the website | * I can track and manage past, current and future students and teachers. As well as, their personal information such as: age, sex, languages, lesson type/frequency and any other preferences  1. Create the database (schema) implementing the Django framework or DBMS (MYSQL/PostgreSQL) 2. Relations created either with Django or the DBMS e.g.    * + - Teacher        - Student        - Parent        - Instrument        - Enrollment        - TermPayment        - Enquiry 3. Added fields e.g.    * teacherID    * fName    * lName    * gender    * dob    * email    * phone 4. Enforced data integrity e.g. dob is of type DateTime 5. Denoted keys and their associations (Primary and Foreign) 6. Stored procedures for most frequent queries 7. Indexes for most frequented relations 8. Yak yak yak   **Acceptance Criteria**  --- | Must | 9 |
| 4 | The Owner of the business | A separate “manage” portal / page | * I can arrange contracts, manage lesson participants and times, reschedule/cancellations and view if instruments have been hired, by whom at what cost/condition.  1. A unique username and password created for the user (owner) via the Django Authentication System and User API 2. The user will have access (permissions) to information set consequently via the User API e.g. Owner has permissions to remove add and remove users e.g. A Teachers record from the database can be deleted if they have resigned 3. User can then log in via the Admin interface provided from the Django framework from the website (address/admin) 4. The user can view information returned from the database via the Admin interface by implementing the Model API 5. Implementing the Model/Object API The user has basic CRUD functionality 6. The user has query options e.g. can view records of hired instruments by which Students from a given date by implementing the Model API   **Acceptance Criteria**   * Test driven development practices, unit tests pass * User understands how to navigate the Admin interface (training) * Communication with the backend (database) for user * Can query records from the database backend | Must | 9 |
| 6 | A Student or Parent | A log in portal / client access page /enrollment | * As a new or existing client, I can view/manage lessons, lesson scheduling/cancellations, lessons preferences and update my personal details online.  1. A login page created for the users Students or Parents 2. The user will have their own unique username and password created by implementing the Django User and Forms API 3. ---   **Acceptance Criteria**  **---** | Must | 8 |